INVESTIGATION OF THE U.S. MARKET FOR TSO AND COMPARABLE SYSTEMS

ABOUT INPUT

INPUT provides planning information, analysis, and recommendations to managers and executives in the information processing industries. Through market research, technology forecasting, and competitive analysis, INPUT supports client management in making informed decisions. Continuing services are provided to users and vendors of computers, communications, and office products and services.

The company carries out co
research. Working closely v
tant issues, INPUT's staff
interpret the research data,
mendations and innovative
needs. Clients receive re
access to data on which an
continuous consulting.

Many of INPUT's professional staff members have nearly 20 years experience in their areas of specialization. Most have held senior management positions in operations, marketing, or planning. This expertise enables INPUT to supply practical solutions to complex business problems.

Formed in 1974, INPUT has become a leading international consulting firm. Clients include over 100 of the world's largest and most techni-
ompanies.

Y-HIT
TSO

Burwen, Michael
AUTHOR

Investigation of the U.S. Market for TSO and Comparable Systems. 2/81

DATE

IT
D

Headquarters
2471 East Bayshore Road
Suite 600
Palo Alto, California 9430
(415) 493-1600
Telex 171407

Los Angeles
4676 Admiralty Way
#401 C
Marina Del Rey, California
(213) 823-1230

lain Street
4
n, Michigan 48170
9-8730

on, D.C.
rth Lynn Street
)
, Virginia 22209
2-2118

UNITED KINGDOM
INPUT, Ltd.
Airwork House (4th Floor)
35 Piccadilly
London, W.1.
England
01-439-4442
Telex 269776

Tokyo
Japan 160
(03) 371-3082

TES

Australia
Highland Centre, 7-9 Merriwa St.,
P.O. Box 110,
Gordon N.S.W. 2072
(02) 498-8199
Telex AA 24434

Italy
PGP Sistema SRL
20127 Milano
Via Soperga 36
Italy
Milan 284-2850

INPUT Planning Services for Management

X-PRO
812

INVESTIGATION OF THE U.S. MARKET FOR TSO AND COMPARABLE SYSTEMS

Prepared For:

HITACHI, LTD.
SOFTWARE WORKS

FEBRUARY 1981

INVESTIGATION OF THE U.S. MARKET FOR TSO AND COMPARABLE SYSTEMS

ABSTRACT

This report contains an analysis of the U.S. market for TSO and comparable systems. Two submarkets are analyzed, namely the "university" and "business" environments. Focus is on the large-scale IBM/MVS market although opportunities in other IBM and non-IBM markets are examined in a cursory fashion. The study also contains an examination of existing products and competitors and gives current market shares.

Y-TSO-02
February 1981

INPUT

INVESTIGATION OF THE U.S. MARKET FOR TSO AND COMPARABLE SYSTEMS

TABLE OF CONTENTS

INVESTIGATION OF THE U.S. MARKET FOR TSO AND COMPARABLE SYSTEMS

LIST OF EXHIBITS

I INTRODUCTION

I INTRODUCTION

A. STATEMENT OF OBJECTIVES

- An objective of this study is to determine the overall characteristics of the TSO and TSO equivalent systems market in the U.S. To do this, we have investigated:

  - The general characteristics of the market.

  - The size and growth rate of the market.

  - Competitive systems and their use.

- The primary objective is to determine overall use of timesharing software in the large IBM CPU market for 1980 and to forecast timesharing usage for 1985.

- Within the overall market, the size of individual segments was determined by:

  - The number of terminals used.

    - Thirty to 80 terminals.

    - Eighty plus terminals.

- The type of user environment.
    - University or laboratory. (Scientific and engineering problem solving.)
    - Corporations. (Business programming and problem solving.)
- The category of mainframe used for the timesharing system.
    - IBM and PCM.
    - Univac.
    - CDC.
    - Honeywell.
    - DEC.
    - Burroughs and NCR.

- To determine the number and type of CPUs used for timesharing, configurations are broken out by:
    - The number and type of CPUs used.
    - The number of terminals used.
    - The operating system used.

- Another objective is to provide information on current suppliers of TSO-type systems being used in the U.S.
    - System name and vendor.

- Purchase or lease terms.

- Any available literature.

- A preliminary evaluation of each competitor.

B. RESEARCH METHODOLOGY

- To study the overall market size and structure, INPUT used:

  - Data from the Computer Intelligence Corporation data base on large IBM and PCM sites.

  - INPUT's existing forecasts of CPU and timesharing market growth.

- To study the vendors of software that competes with TSO, INPUT conducted interviews with seven vendors to determine:

  - The number of product installations.

  - Market growth rates.

  - Marketing strategies.

  - Vendor strengths and weaknesses.

  - Assessment of the timesharing software market by the competing vendors.

- To study the effect of other mainframe vendors on the IBM timesharing market, INPUT used data from the CIC data base and interviewed five mainframe vendors to determine:

- Market size and growth.

- Marketing strategies in the IBM market.

- The ability to compete.

- Because preliminary data showed a definite trend to the use of mini- and microcomputers for interactive problem solving, selective interviews with vendors of these systems were conducted to determine:

  - The long-term effect on large systems used for timesharing.

  - Marketing strategies.

  - Anticipated growth rates.

- A few selected users were interviewed to assist in determining:

  - Competing software quality.

  - Required vendor characteristics.

  - Minimum features for software product viability.

II EXECUTIVE SUMMARY

II EXECUTIVE SUMMARY

A. CONCLUSIONS

- The market for TSO-type timesharing software on very large IBM mainframes in the U.S. is excellent.
  - The market will grow substantially, from 1,350 user installations in 1980 to 3,100 user installations in 1985, as shown in Exhibit II-1.
  - The greatest growth will occur among the largest users (more than 80 terminals).
  - The business market will offer greater potential than the university/-laboratory market.
- The market growth will not slow down during the forecast time period.
  - New users are moving up into the MVS/timesharing environment from smaller systems.
  - Existing users of large IBM and PCM computers are adding additional systems, or converting batch systems for on-line problem solving and programming.

EXHIBIT II-1

TSO AND EQUIVALENT
INSTALLATION MARKET FORECAST FOR
IBM AND PCM (3032 AND LARGER)

MARKET	30-80 TERMINALS		81+ TERMINALS		TOTAL	
	1980	1985	1980	1985	1980	1985
UNIVERSITIES/ LABORATORIES	70	141	224	557	294	698
BUSINESS CORPORATIONS	253	466	804	1,918	1,057	2,384
TOTALS	323	607	1,028	2,475	1,351	3,082

- In the U.S., public data networks (Telenet, Tymnet) have not had a significant impact on the large timesharing market, particularly with IBM mainframe users.

    - In the IBM market, asynchronous terminals, which are easily supported by the public data networks, are less likely to be used than 3270s.

    - Users with large networks have plans to convert to SNA or, in some cases, to use other private network products.

- There are several well-established TSO competitors in the market today.

    - Most products are well-advertised and promoted.

    - A few vendors have been successful by targeting specific market niches.

- There are several reasons why TSO users are reluctant to change systems:

    - Uncertainties of conversion:

        - Time involved.

        - Cost.

        - Added burden to retrain users on the use of a new system.

        - Uncertainty of the acceptance of a new product by people accustomed to TSO.

    - Anticipation of new IBM software.

    - Reluctance to use anything that is not directly supported by IBM.

- Not perceived as important enough to justify a change.

- The price range for timesharing software is $15,000 to $30,000 for a perpetual license.

  - An additional maintenance fee is charged after the first two years of use.

  - The annual maintenance fee is typically $1,000 to $1,500 but can be as high as 15% of the license fee.

- A new product entering the market would not need to have all the features of other competing products.

  - Lack of features would limit market share but would not preclude market entry.

  - As a product evolves over time, users will pay additional fees for added capabilities.

- TSO is recognized by most users to be an inefficient system that lacks several important functions. When users turn to alternatives, they are looking for:

  - More efficient use of the CPU.

  - User friendliness.

  - Full screen and file editing.

  - Foreground processing of programs.

  - More efficient disk storage utilization.

  - Easy document preparation for all users.

- TSO has a poor reputation among many users. Typical negative comments about TSO are:

  - "Very slow response time."

  - "Uses too much CPU resources."

  - "Complicated sign-on and command structure."

  - "Output processing is poor."

  - "Difficult to get acceptance from programmers who have used other systems."

- Because of TSO's reputation as an inefficient and difficult system to use, first-time timesharing users are often open to alternative products.

- Japanese companies are not known in the U.S. as potential software vendors.

- The key to success is not technical excellence. Rather, it is broad sales coverage, effective promotion and advertising, and a product that is very reliable.

- Developing a direct sales force to cover the entire U.S. would be too expensive unless the Hitachi timesharing product is:

  - Aimed at a specific market niche.

  - Accompanied by other products so that a salesman could offer several products on each sales call.

B. RECOMMENDATIONS

- To most effectively sell Hitachi timesharing software in the U.S., INPUT recommends affiliating with:

  - A software broker offering nationwide coverage.

  - An existing software vendor that does not now offer a timesharing software system.

- Before beginning a sales campaign for Hitachi software in the U.S., it will be necessary to have already completed several preparatory work phases:

  - The product must be thoroughly tested in a variety of customer sites.

  - The product must be well documented. All literature must be easily understood by users familiar with IBM documentation.

  - High-quality promotion materials must be developed. These can be expected to receive wide distribution before many prospects pay to become users.

- The Hitachi software product should be listed in the software directories that many users refer to when selecting a product. These include:

  - ICP Quarterly.

  - Datapro.

  - Auerbach.

- Market entry for a new product in the U.S. market should begin as soon as possible. There are already several successful products, and the competing vendors are becoming better entrenched by increasing their market share and product mix.

- Because we do not know the features and specifications of the Hitachi timesharing software product, INPUT is not able to recommend which market segment, either user type or user size, Hitachi should concentrate on.

- For further in-depth study, INPUT recommends that Hitachi investigate the following products and vendors:

  - <u>ROSCOE from Applied Data Research</u>: ROSCOE because it is the most successful competitor despite having fewer features than its competitors. As important as the product is, ADR itself should be studied as a highly successful software vendor.

  - <u>Tone-3 and Tone-4 from Tone Software</u>: Tone-3 because of its success in a particular market niche; Tone-4 because it competes on a feature basis with the latest upgrades to TSO.

  - <u>Wylbur and Superwylbur from On-Line Business Systems</u>: Wylbur because it has been the model for user-friendly systems, offering excellent user features. Yet, despite having a good reputation, these systems have never achieved the market penetration of ROSCOE and Tone.

C. FUTURE TRENDS AND CHARACTERISTICS IN USAGE OF TIMESHARING SYSTEMS

- For at least ten years now, there has been an almost universal belief that on-line access to program development tools results in greater programming productivity.

- In INPUT's mail survey, the largest group of respondents named on-line interactive program development as the single most successful factor for productivity improvement.

- This belief is supported primarily by logic.

  - It is reasonable to assume that the periods of waiting for batch outputs of trial runs are largely unproductive.

  - Even when the programmer attempts to multiplex the time by working on program B while waiting for program A's output, the mental "gear shifting" from one line of thought to another probably degrades productivity.

- There is a nagging suspicion that excessive on-line access can be detrimental to productivity by encouraging sloppy work. This suspicion was expressed by a number of people INPUT interviewed. It has also been reported in other industry literature.

  - However, it can also be reasonably argued that this fault is not inherent in the interactive environment and can be avoided by establishing discipline through appropriate management practices.

- The elements of the interactive development environment are by now common to many varieties of hardware and to many software monitors. They include:

  - A mechanism for controlling the communications lines to a number of terminals.

  - An editor to allow terminal operators to enter, modify and examine text, especially source code and listings.

- A command language to enable terminal users to invoke assembly, compilation, link editing and execution of programs, often with dynamic tracing capabilities and symbolic debugging features.

- A "HELP" menu to guide the uninitiated in the details of the interactive system.

- When these elements are reentrant, the efficiency of the system is greatly enhanced by the ability to service multiple editing operations with a single copy of a reentrant editor.

- TSO and CMS are the two best-known on-line systems from IBM.

  - TSO ("Time Share Option") is designed to reconcile the batch orientation of IBM operating systems, especially MVT, SVS and MVS, with the need to service interactive program development and testing. TSO combines the functions of a communications monitor, controlling a large number of users' terminals, with such services as interactive data set creation, text editing, compilation and test runs. To the operating system, TSO appears to be just another "job."

  - CMS (Conversational Monitor System) is the "native" operating system of VM/370, a system designed to permit execution of many different operating systems on a single CPU. CMS services a single-terminal user with facilities similar to TSO. By running multiple "CMS machines," a VM-controlled CPU can create a multiterminal interactive environment similar to TSO.

- More recently developed operating systems from other vendors do not need artifices like TSO or CMS, because they are inherently designed to support an interactive environment.

- The TOPS10 and TOPS20 operating systems for the DEC PDP-10 and -20 (now called DEC systems 10 and 20) are examples of an operating system designed to support both batch and interactive workloads.

- The operating system for the Wang 2200 and VS computer lines are entirely interactive-oriented, and assume that each user has a CRT workstation. Batch workload is handled by treating the batch input and output as if they come from, or are destined to, a dummy or simulated interactive terminal.

• The underlying, common problem of all existing tools and aids is their ad hoc nature; they each address a specific area independently of other related areas. For example:

  - Existing requirements languages have no interface to the detail design and implementation phase.

  - Structured design and analysis methodologies have little or no transition mechanism into coding.

  - Structured design and analysis methodologies address the entire life cycle, but only as a "checklist"; i.e., without tools to address the substance of the life cycle activities.

• Another major problem is the scarcity and relative ineffectiveness of techniques, tools and aids to influence the maintenance activity directly.

  - Useful tools not now commercially available would be:

    . A global cross-reference generator, operating across the entire application system of main program, called subroutines, job control statements, utility parameter lists, etc.

        - A mechanism to retroactively populate the data dictionary (related to the cross-reference generator).

        - A global documentation generator/editor that would facilitate structured, "reusable" documentation.

        - A regression test harness or driver.

    - The absence of these tools leaves a rather big gap in productivity improvement, since INPUT estimates that the majority of the life cycle costs, at least of major projects, are due to maintenance rather than development activities.

- What is clearly needed is a system to integrate the automation of all life cycle activities into a uniform, compatible and communicating set of techniques and tools.

- Using the existing state of the art as a base, without requiring any revolutionary breakthroughs, it is possible to visualize such a system, and to list its major features and characteristics.

    - It will have an interactive "workbench" to serve the needs not only of coders, but also of system specifiers, system designers and maintenance personnel.

    - The system specifiers and designers, working interactively via their unique "workbench," will accumulate the specifications and overall design of the system, cast in a "requirements language."

    - The requirements language processor will store the elements of the system specifications and design in the same data base and data dictionary that will eventually be used for the operation of the system being designed.

- The requirements language processor will perform consistency checks and produce documentation to serve the rest of the system's life cycle.

- The requirements language processor will produce one or several alternate "trial" designs, and will forecast their performance under a range of operating assumptions.

- Specifiers and designers will think in terms of a uniform, company-wide structured design and analysis methodology.

- Detail designers and coders, supported by their own unique "work-bench," will convert the procedures and data designated in the specifications phase into executable code, using a very high-level language supported by the appropriate code generator.

- They will be able to search a large library of previously developed standard functions and skeletons by context, to determine which of the existing modules and skeletons already satisfy portions of the new design.

- Some end user requirements will not even reach the DP department, because the installation's data base management system, with its associated query/retrieval language, will allow them to create mini-applications without assistance.

- Throughout the entire process, the activities of all personnel will be guided by a checklist provided by a system development methodology.

- Progress reports and personnel/resource loading data will be automatically produced by the system's automated components. For this purpose, these components will be able to exchange data.

- Documentation will be produced automatically from the data in the requirements data base and data dictionary.

- Both planned and unplanned enhancements to the system will be handled as "projects," although they will not require the complete SDM checklist.

- There will be an automated "version control" system that permits binary modules to be unambiguously identified with the source code and design versions that produced the binary modules. This system will also control what goes into the "production" version of the system.

- Note that all of the above capabilities already exist in some automated form or another.

  - What is missing is the "glue" that binds the individual techniques into a uniform, compatible, communicating system.

  - The other missing element is the automation of program validation and checking.

- Clearly, future progress must then proceed along two lines of attack:

  - Developing integrated systems of uniform, communicating tools.

  - Developing effective, simple tools to automate program validation and checking.

III MARKET POSITIONS AND TRENDS

III MARKET POSITIONS AND TRENDS

A. THE LARGE CPU-MVS ENVIRONMENT

- Based on data from the Computer Intelligence Corporation (CIC) data base and corroborated from other sources, we have determined that there are, as of year end 1980, 2,300 computers installed in the U.S. in the size range under study, as shown in Exhibit III-1.

- Of these 2,300 CPUs, 1,975 of them are using the MVS system.

- Of the 1,975 CPUs using MVS, 1,677 are used as dedicated or part-time timesharing systems.

- Of the 1,677 CPUs supporting timesharing, 1,477 support 30 or more terminals.

- Of those 1,477 CPUs, 1,351 are used in either a business or a university/laboratory environment.

- It is these 1,351 CPUs that are discussed in the following sections.

EXHIBIT III-1

IBM AND PCM
1980 CPU POPULATION IN U.S.

CATEGORY	PERCENT	NUMBER OF CPUs, 1980
TOTAL OF VERY LARGE CPUs (168s, 3032s, 3033s AND AMDAHL)	100%	2,300
USING MVS	86	1,975
USING TIMESHARING FULL OR PART-TIME	73	1,677
SUPPORTING 30 OR MORE TERMINALS	64	1,477
(30-80 TERMINALS = 353)	15	
(81 OR MORE TERMINALS = 1,124) *	49	
USED IN BUSINESS OR LABORATORIES	59	1,351
BUSINESS = 1,057	46	
LABORATORY OR UNIVERSITY = 294**	13	

*LESS THAN 30 TERMINALS: 200
**GOVERNMENT: 126

B. THE UNIVERSITY/LABORATORY ENVIRONMENT

- As shown in Exhibits III-2 and III-3, IBM's TSO is the dominant timesharing software system used in the university/laboratory environment.

- Discussions with users in this environment indicated that by the time a large user is running under MVS, he is usually committed to using IBM software, such as TSO.

- Perhaps surprisingly, there are many MVS users in the university/laboratory environment.

    - IBM is still the dominant supplier, and MVS is the premier operating system for all large-scale CPUs in any environment.

    - MVS is the standard for Amdahl CPUs.

    - Universities use MVS partly because their students are being trained for positions in businesses which are likely to have large IBM CPUs using MVS.

    - MVS will be required for future CPU products from IBM.

- There is not a significant difference between the attitudes of university/-laboratory people and those in business who use very large IBM systems. In both cases, the systems are managed by DP professionals.

    - In the case of the university, if end users demand and pay for other systems, the DP department will be more likely to accede to these demands than in the business environment.

EXHIBIT III-2

TSO MARKET SHARES IN UNIVERSITIES/LABORATORIES (30-80 TERMINALS)

VENDOR	SOFTWARE SYSTEM	ESTIMATED NUMBER OF SITES YEAR 1980	MARKET SHARE PERCENT AS OF YEAR 1980
IBM	TSO	50	71
ADR	ROSCOE	13	19
CULLINANE	INTERACT	0	-
ON-LINE BUSINESS SYSTEMS	WYLBUR AND SUPERWYLBUR	6	9
TONE SOFTWARE	TONE-3 AND TONE-4	0	-
McGILL UNIVERSITY (MARKETED BY IBM)	MUSIC	0	-
OTHERS	-	1	1
TOTAL	-	70	100%

EXHIBIT III-3

TSO MARKET SHARES IN UNIVERSITIES/LABORITORIES
(OVER 80 TERMINALS)

VENDOR	SOFTWARE SYSTEM	ESTIMATED NUMBER OF SITES YEAR 1980	MARKET SHARE PERCENT AS OF YEAR 1980
IBM	TSO	184	82%
ADR	ROSCOE	18	8
CULLINANE	INTERACT	2	1
ON-LINE BUSINESS SYSTEMS	WYLBUR AND SUPERWYLBUR	16	7
TONE SOFTWARE	TONE-3 AND TONE-4	2	1
McGILL UNIVERSITY (MARKETED BY IBM)	MUSIC	0	-
OTHERS	-	2	1
TOTAL	-	224	100%

- Market share data shows that longevity in the marketplace is the key to market share. Both ROSCOE and Wylbur have been available for six or more years.

- In the university/laboratory - but not MVS - environment, users are more likely to experiment with and become users of competitive software. An example would be the limited success of MUSIC that runs only under VM.

C. THE BUSINESS CORPORATION ENVIRONMENT

- As shown in Exhibits III-4 and III-5, the market share data do not drastically differ between the business environment and the university environment.

- TSO is the clear market leader, with ROSCOE and the various versions of Wylbur well behind.

- Again, product quality does not guarantee immediate market share. Vendors take several years to build up to a 10% share.

- It is significant that no generalized timesharing system is clearly a leader in one environment rather than the other. Selling in one environment is as difficult or as easy as the other.

D. THE TOTAL LARGE CPU ENVIRONMENT

- Exhibit III-6 shows the market share data for all of the major competing software systems in the large CPU environment.

EXHIBIT III-4

TSO MARKET SHARES IN BUSINESS CORPORATIONS (30-80 TERMINALS)

VENDOR	SOFTWARE SYSTEM	ESTIMATED NUMBER OF SITES YEAR 1980	MARKET SHARE PERCENT AS OF YEAR 1980
IBM	TSO	195	77
ADR	ROSCOE	30	12
CULLINANE	INTERACT	2	1
ON-LINE BUSINESS SYSTEMS	WYLBUR AND SUPERWYLBUR	21	8
TONE SOFTWARE	TONE-3 AND TONE-4	3	1
McGILL UNIVERSITY (MARKETED BY IBM)	MUSIC	0	-
OTHERS	-	2	1
TOTAL	-	253	100%

EXHIBIT III-5

TSO MARKET SHARES IN BUSINESS CORPORATIONS (OVER 80 TERMINALS)

VENDOR	SOFTWARE SYSTEM	ESTIMATED NUMBER OF SITES YEAR 1980	MARKET SHARE PERCENT AS OF YEAR 1980
IBM	TSO	649	81%
ADR	ROSCOE	78	10
CULLINANE	INTERACT	8	1
ON-LINE BUSINESS SYSTEMS	WYLBUR AND SUPERWYLBUR	51	6
TONE SOFTWARE	TONE-3 AND TONE-4	9	1
McGILL UNIVERSITY (MARKETED BY IBM)	MUSIC	0	-
OTHERS	-	9	1
TOTAL	-	804	100%

EXHIBIT III-6

MARKET SHARES OF TIMESHARING SOFTWARE - VERY LARGE SYSTEMS (IBM/168, 3032, 3033; AMDAHL)

SOFTWARE SYSTEM	OPERATING SYSTEM					MARKET SHARE (PERCENT)
	VSI	SVS	MVS	OTHERS	TOTAL	
TSO	0	38	1,340	0	1,378	74%
ROSCOE	22	15	174	14	225	12
INTERACT	3	0	12	0	15	1
WYLBUR	5	0	26	1	32	2
SUPERWYLBUR	0	2	92	2	96	5
TONE-3	45	0	0	0	45	2
TONE-4	0	0	18	0	18	1
MUSIC	0	0	0	12	12	1
OTHERS	4	2	15	9	30	2
TOTAL	79	57	1,677	38	1,851	100%

- When broken down by operating system rather than by user classification or number of terminals, certain products are only sold into a limited O/S environment (Tone 3 and Tone 4).

- Further, it can be seen that some products have done exceptionally well in a particular environment (ROSCOE with SVS or Superwylbur with MVS).

- Remember that Exhibit III-6 includes CPUs with less than 30 terminals and governmental users.

E. USE OF NON-IBM CPUs

- Analysis of the CIC file showed that in only 3% of the identified sites were large mainframes, such as CDC, Univac 1100 or DEC-10, used alongside IBM or PCM CPUs. Since all of these systems support timesharing, it is not possible to separate timesharing usage from nontimesharing usage, as shown in Exhibits III-7 and III-8.

- All of these competing CPUs were shown in locations running MVS.

- Fifty-five percent of those users who have both a non-IBM CPU and an IBM or PCM CPU are not TSO users. This is approximately 20% higher than all non-TSO users.

F. THE MEDIUM TO LARGE CPU TIMESHARING MARKET

- Although not formally a part of this study, it was necessary to analyze the market shares of TSO and its competitors in the medium to large CPU range, as shown in Exhibit III-9, in order to develop data for the large to very large CPU range, as shown in Exhibit III-6.

EXHIBIT III-7

TSO (MVS) AND TSO COMPETITORS -
U.S. MARKET FORECAST FOR UNIVERSITIES/LABORATORIES

MAINFRAME VENDOR	ESTIMATED NUMBER OF CPUs 30-80 TERMINALS			ESTIMATED NUMBER OF CPUs 81+ TERMINALS		
	1980	1985	AAGR (PERCENT)	1980	1985	AAGR (PERCENT)
IBM + PCM 3032 + LARGER	70	141	15%	224	557	20%
CDC	14	19	5	52	76	8
DEC	7	10	8	42	51	4
HONEYWELL	3	6	15	29	78	22
UNIVAC	19	15	11	32	59	13
IBM SITES USING DEC, CDC, HONEYWELL, ETC.	4	6	10	7	11	10

EXHIBIT III-8

TSO (MVS) AND TSO COMPETITORS -
U.S. MARKET FORECAST FOR BUSINESS CORPORATIONS

MAINFRAME VENDOR	ESTIMATED NUMBER OF CPUs 30-80 TERMINALS			ESTIMATED NUMBER OF CPUs 81+ TERMINALS		
	1980	1985	AAGR (PERCENT)	1980	1985	AAGR (PERCENT)
IBM + PCM (3032 + LARGER)	253	466	13%	804	1,918	19%
CDC	3	4	5	18	25	7
DEC	0	0	-	5	7	7
HONEYWELL	7	12	12	76	160	16
UNIVAC	15	22	8	80	123	9
IBM SITES USING CDC, DEC, HONEYWELL, ETC.	8	14	12	23	44	14

EXHIBIT III-9

MARKET SHARES OF TIMESHARING SOFTWARE -
MEDIUM-LARGE SYSTEMS (IBM/158, 3031, 155, 165; AS6)

SOFTWARE SYSTEM	OPERATING SYSTEM					MARKET SHARE (PERCENT)
	VSI	SVS	MVS	OTHERS	TOTAL	
TSO	0	56	997	0	1,053	58%
ROSCOE	49	21	319	36	425	24
INTERACT	21	1	28	0	50	3
WYLBUR	9	1	16	2	28	2
SUPERWYLBUR	6	2	44	7	59	3
TONE-3	120	0	0	0	120	7
TONE-4	0	0	14	0	14	1
MUSIC	0	0	0	8	8	-
OTHERS	7	2	17	14	40	2
TOTAL	212	83	1,435	67	1,797	100%

- This is obviously a good market also, although it has two major differences when compared to the very large CPU market:

  - The number of terminals supported per system is considerably smaller.

  - The growth rate for both CPUs in general and for timesharing in this size range is less than in the very large CPU range.

IV SYSTEM CONFIGURATIONS

IV SYSTEM CONFIGURATIONS

A. CPUs PER SITE

- Based on data from the Computer Intelligence Corporation (CIC) data base of installed CPUs, Exhibit IV-1 indicates how the use of MVS and TSO varies with the number of CPUs per site.

- Over half (52%) of all sites using very large IBM and PCM CPUs have only one CPU. But these sites account for only 27% of the CPUs in this size range.

    - Eighty-one percent of these sites report they are using MVS.

    - Forty-seven percent of these MVS users report using TSO.

- Sites with two of these CPUs represent 28% of all sites and have 30% of the CPUs.

    - Eighty-nine percent of two CPU sites report using MVS.

    - Sixty-one percent of these MVS users report using TSO.

- Sites with three or more large CPUs represent only 20% of all sites but have over 43% of the CPUs.

EXHIBIT IV-1

ANALYSIS OF LARGE CPU USERS
BASED ON CPUs PER SITE

CPU USERS	1 CPU (PERCENT)	2 CPUs (PERCENT)	3 OR MORE CPUs (PERCENT)
SHARE OF TOTAL SITES	52%	28%	20%
SHARE OF CPUs	27	30	43
PERCENT OF SITES IN THIS CATEGORY RUNNING MVS	81	89	92
PERCENT OF SITES RUNNING MVS THAT ALSO USE TSO	47	61	65
SHARE OF ALL TSO USERS	41%	33%	26%

- The average number of CPUs used by these sites is four, although there are a few sites in the CIC data base with eight or more CPUs.

- Ninety-two percent of these sites use MVS.

- Sixty-five percent of the MVS users also run TSO.

- While available data are too sketchy to be conclusive, use of competitive timesharing systems is apparently less frequent among these multi-CPU users.

B. USE OF OTHER MAINFRAMES

- As reported in Chapter III, only 3% of all sites reported using a large non-IBM-compatible CPU along with the large IBM and PCM CPUs in this study.

- Two percent reported using Burroughs, NCR, Univac or Honeywell CPUs. These are primarily used in the business environment.

- Only 1% of users reported also using CDC, CRAY or large DEC CPUs, and these are all in the university/laboratory environment.

  - The largest reported timesharing user had over 350 terminals on a CDC computer.

C. NUMBER OF TERMINALS

- As shown in Exhibit IV-2, users of large CPUs reported using a large number of terminals.

EXHIBIT IV-2

ANALYSIS OF NUMBER OF TERMINALS USED IN TIMESHARING ENVIRONMENT BY LARGE CPU USERS

CPU USERS	<30 TERMINALS (PERCENT)	30-80 TERMINALS (PERCENT)	>80 TERMINALS (PERCENT)
SHARE OF TOTAL SITES	11%	23%	66%
PERCENT OF SITES RUNNING MVS	85	82	87
PERCENT OF SITES RUNNING MVS THAT ALSO USE TSO	45	51	58
SHARE OF ALL TSO USERS	9%	20%	71%

    - The terminals counted in this study were 3270 types, other CRTs and teleprinters. Industry-specific and RJE terminals were excluded.

- Sixty-six percent of all sites reported using more than 80 terminals.

    - These sites also have the largest percentage of MVS and MVS/TSO users.

    - Most importantly, 71% of all sites reporting using TSO are in the more-than-80-terminal category.

- The 30- to 80-terminal category represents 23% of all sites.

    - This category reports the lowest usage of MVS - 82%.

- Only 11% of the large user sites reported using less than 30 terminals.

- The maximum number of terminals used on general-purpose IBM systems appears to be in the 250-300 range.

    - Some CDC users reported as many as 350 terminals.

    - Special-purpose systems (e.g., airline reservations) go much higher.

D. USER CLASSIFICATION

- Seventy-two percent of all sites reported by CIC are classified as business corporations according to their Standard Industry Classification (SIC) code.

- Twenty percent of the sites are classified as universities, other schools or laboratories.

- Eight percent are government-related installations.

E. COMMENTS ON CIC DATA

- The CIC data base contains user information on a site basis. CIC does not claim to list 100% of all sites, so other data and INPUT's best judgement have been used in developing all figures shown in this report except those in IV-A through IV-D above.

- The CIC data base is most accurate on CPU data. It becomes less accurate, or less up-to-date, the further from the CPU it gets. For instance:

  - Reported usage of TSO would be less complete than for MVS.

  - Terminals listed in the data base would not be as accurate as data on disks or CPUs.

- CIC has only recently begun systematically to capture data on software packages that compete with TSO. Therefore, market share data was necessarily derived from vendor interviews and other sources available to INPUT.

- These comments do not diminish the value of the CIC data. Rather, they only point out that this data base is a very useful tool that must be used by experienced analysts. CIC has the best file of its kind available.

V TIMESHARING PRODUCTS AND VENDORS

V TIMESHARING PRODUCTS AND VENDORS

A. ALL COMPETITION

- The total market for on-line programming and problem solving becomes more complex over time. Today, the viable competitors are:

  - Timesharing service companies.

  - In-house timesharing based on large CPUs - the object of this study.

  - Special-purpose, large CPUs.

  - Minicomputers.

  - Microcomputers.

- The timesharing services companies will not be discussed as part of this study. They do not represent a market for new timesharing software products.

B. SOFTWARE PRODUCTS FOR IBM-TYPE COMPUTERS

- TSO: the timesharing option of IBM's operating systems.

- Known as an inefficient and difficult system to use.

- Despite its drawbacks, TSO is by far the most widely used timesharing software package.

- IBM has been attempting to combat TSO competitors by introducing TSO enhancements. The most important of these is the Structured Programming Facility (SPF) which offers some of the features of competitors such as Wylbur.

● <u>ROSCOE</u>: the vendor is Applied Data Research (ADR).

  - ADR claims that there are over 900 installations worldwide.

  - In the U.S., ADR has nine sales offices with an average of five salespersons and at least one SE in each.

    . ADR stresses ease of use, compared to TSO, as the major selling point.

    . The largest known ROSCOE network supports over 250 terminals; the average is 40-50 terminals.

    . ADR states that the largest competitor today is the Structured Programming Facility (SPF) of TSO.

    . Many ROSCOE users also use TSO.

● <u>Wylbur and Superwylbur</u>: the vendor is On-Line Business Systems (OBS).

  - Wylbur was originally developed by Stanford University. OBS was formed in 1976 to market Wylbur.

- The Superwylbur package was developed by Optimum Systems, Inc. (OSI) out of public domain versions of Wylbur.

- OSI has sold its software business to OBS.

- There are over 200 installations of Wylbur and Superwylbur.

- Wylbur is sold on the basis of greater capability as well as ease of use and efficiency.

- SPF is now a major competitor of Wylbur.

- OBS feels that word processing systems are now providing competition since that was a major capability of Wylbur.

• Interact: the vendor is Cullinane Software.

  - This is also a derivative of the original Wylbur.

  - There are over 80 installations worldwide.

  - The average system supports 75 terminals. The largest known installation is approximately 175 terminals.

  - Interact does not provide foreground execution. It is only a programming aid.

  - Interact is sold almost exclusively where other Cullinane products are sold.

• Tone-3 and Tone-4: the vendor is Tone Software.

  - Tone-3 is derived from public domain TSO. It was developed to run in a VSI environment.

  - Tone claims over 200 installations of Tone-3 worldwide.

  - Tone uses only two salespersons in the U.S.

  - Tone-4 is an outright TSO replacement, MVS only.

  - Designed to cut overhead, it has few enchancements.

  - It requires little technical support or training because it closely copies TSO.

- <u>MUSIC</u>: the vendor is McGill University, Montreal, Canada.

  - MUSIC is developed specifically for a student environment.

  - It contains its own operating system - runs under VM/370.

  - Two-thirds of the approximately 40 installations are in schools.

  - McGill is moving slowly to make this a moneymaker.

  - The main benefit claimed is reduced hardware requirement.

  - Most installations support a non-professional programmer environment.

  - MUSIC is sold through IBM as a contributed program product.

- All these software vendors felt the market was growing at 25-50% per year.

- Pricing is not a major issue in this market. All are sold mostly on full payout leases at $800-1,000 per month for 24 months. None has a full purchase price of over $30,000, or less than $17,500. Rentals are offered by all vendors, but very few installations choose this option; it is primarily for government agencies that can only get funding for a limited period of time.

- All vendors see IBM as the major competitor in the MVS environment. This is an indication that the market has not reached a maximum level of penetration by non-IBM competitors.

- There is another category of software products that is sometimes not clearly distinguished from timesharing-type systems. These are data communications monitors. They compete more with CICS or with IMS/DC than with TSO.

  - These products are sometimes used to develop limited systems that do provide on-line services such as remote program execution.

  - Some of the products in this category are:

    - Intercomm.

    - Task/Master.

    - Com-Plete.

    - Environ/I.

    - Shadow II.

  - Users of these products who do not have a timesharing software system are usually the easiest prospects for selling timesharing software.

C. COMPUTER SYSTEMS

- All vendors of computer systems today offer timesharing software. All of the vendors interviewed were certain that they offered a more cost-effective system with TSO than IBM. Despite this, not one has a strategy to try to displace such IBM systems.

- Digital Equipment Corporation (DEC): the DEC product that would compete directly with a large IBM system is the SYSTEM-10. The operating system is a timesharing system. The software cannot be broken into separate pieces.

  - The DEC System-10 is more than 12 years old.

  - DEC is planning a performance upgrade for the System-10 along with increased marketing effort.

  - The System-10 has done very well in the university/laboratory environment, poorly in the business world.

  - DEC people know of very few cases where a System-10 is located at an IBM site.

- Control Data Corporation (CDC): CDC claims to have over 500 installations of its large CPUs (6600, 7600, Cyber-175, -176 and -205) that use CDC's Interactive Facility (IAF) software option.

  - CDC systems are successful when pure numeric problem solving is required.

  - The average system supports over 100 terminals; one as many as 350.

  - CDC will aggressively compete with IBM for new installations; it will not target IBM accounts in an attempt to displace the IBM system.

  - The main area of competition is minicomputers. Labs have a difficult time getting large government grants for systems; several smaller systems go through more easily on separate budgets.

- Sperry Univac: the Univac 1100 family of large computers has been successful in both the business and laboratory environments.

- Univac has a large product family so that a user can select from a compatible family of processors starting with systems similar in power to a 3031 and going all the way up to the power of a 3081.

- Univac has only one operating system for the 1100 family, now known as 1100 O/S.

- There are three levels of timesharing software available with 1100 O/S:

  . Conversational Timesharing System (CTS) is the most frequently used, with over 600 installations. It is similar in function to TSO, but easier to use.

  . Demand Processor is the lowest-level, easiest-to-use system. There are approximately 500 installations.

  . High Volume Time Sharing (HVTS) offers CTS-level capability but much better CPU efficiency. The system will easily support 300-500 terminals. There are still less than 10 HVTS users.

● Honeywell: the MULTICS timesharing system that Honeywell acquired with the GE product line 10 years ago keeps Honeywell in the ranks of major timesharing vendors.

  - The MULTICS system was originally developed at M.I.T. with a government grant. (The project was related to the ARPANET project.)

  - MULTICS is used by many government installations today.

  - System security is one of MULTICS's strengths, the reason for its success in military and other government agencies.

  - Honeywell, like CDC, will aggressively go after new business but does not actively attempt to displace IBM systems.

- Burroughs and NCR: these two companies offer timesharing software as part of their product line. Neither has been successful with the product in a pure timesharing environment. Each is most successful supporting business applications on-line, such as banking. Neither should be considered as a possible competitor in the TSO environment.

D. MINICOMPUTER SYSTEMS

- These systems from DEC, HP, Data General, Prime and many others are seldom used today in a non-interactive manner when they are used for programming or problem solving. The software systems assume interactive use and even make batch processing comparatively difficult.

  - Minicomputers are the most significant factor in the university/laboratory timesharing market.

  - Minicomputers are easy to justify in individual department or grant budgets.

  - Minicomputers have user-friendly systems compared to large computer software systems, particularly TSO.

  - Minicomputers offer more than enough computer power for most applications. If not, get a larger mini, use a timesharing service or run batch on a large system. Only as a last resort would a user acquire or share a large CPU.

- The growth of the minicomputer market does not indicate the demise of large timesharing systems. Rather, it just means that many new applications will be implemented slowly, if at all, on the large systems.

- Further, until practical software exists to control a distributed data base, large centralized systems will be necessary for many organizations dependent on the use of massive data volumes.

E. MICROCOMPUTERS

- These personal or very small business computers must be mentioned because they are already a viable alternative to large CPUs.

  - A large San Francisco-based bank is acquiring 3,000 Apple systems faster than it is adding 3270-type terminals. Over 1,000 systems will be installed in 1981. None of these is ordered through the data processing department. Dissatisfaction with TSO among the users, both its difficult use and its cost, is the major incentive.

  - A nationwide distribution company has ordered over 2,000 systems for its sales force. Each salesperson will have their own system for territory management. The plan to issue portable terminals and put the applications on central computers was judged too costly to implement and operate.

- Many micros already are used for word processing - one of the strong points of several timesharing software systems.

- To increase programmer productivity, some installations are letting programmers use micros at home. Edited and tested programs on modules are submitted via the on-line system only when completed.

- Microcomputers will have an effect much like minicomputers. They will slow but not stop the growth of large, centralized computer systems. The effect of micros will be felt as much in the business environment as in the university/laboratory.

F. OTHER IBM PRODUCTS

- While the timesharing software market has provided a large market for alternatives to TSO, IBM itself has been slow to meet the need.

- The most successful IBM alternative has been the VM/370 operating system with the Conversational Monitor System (CMS). The VM/CMS environment is very common in the medium-size computer range because it allows users to run their older DOS systems simultaneously as they upgrade to and run OS (MVS) systems.

- Despite users having TSO as part of MVS, some prefer to stay with CMS because it is so much easier to use than TSO.

- IBM now offers added facilities known as Structured Programming Facility (SPF) for both TSO and CMS. SPF permits such functions as browsing through program files, editing, assembling and testing directed through a terminal.

- As indicated by both vendors and users, SPF provides the major competition for those competing vendors.

- As noted above, IBM is even willing to market a competitive product such as MUSIC. They don't push MUSIC outside of the university environment, but in that environment, it gives them a product that better competes against DEC and CDC.

VI ANALYSIS OF SUCCESSFUL MARKET AND PRODUCT STRATEGIES

VI ANALYSIS OF SUCCESSFUL MARKET AND PRODUCT STRATEGIES

- In the process of interviewing vendors, we attempted to determine the key elements of their success in the market. This was carried out further in the interviews with users.

A. PRODUCT FEATURES

- This study demonstrated what had been determined before in studies of other software products. Lack of features does not prevent the sale of a product, it only limits market share or slows growth.

- Large users who do seek an alternative to TSO will look for features such as full screen edit, foreground processing and document preparation (WP). To move away from TSO almost demands that they get more functions.

- Secondly, users changing from TSO will look for better utilization of resources - always CPU, sometimes disk.

- Thirdly, they want a more user-friendly system, which all of the vendors claim to offer.

- First-time users will buy anything. They have usually heard of the inefficiency of TSO and welcome an alternative. Only if they are dependent on IBM for systems support, an unlikely situation in a TSO installation, will TSO be the first choice.

- Of course there are a few cases where the user buys IBM and nothing else. This is more common in the insurance industry than in any other.

- Therefore, for a system supporting up to 80 terminals, having all the features of SPF is not necessary. A plan to add functions in the future will add credibility but should never be over-emphasized.

B. SALES COVERAGE

- If you have a unique market niche into which only your product fits, as Tone did with its TSO version for VSI sites, a small salesforce with a telephone will suffice.

- When competing head-to-head for most sales, broad coverage is required. ADR has over 50 salespersons in the U.S., Cullinane has almost 25.

- On the other hand, Wylbur and Superwylbur, with more features than ROSCOE or Interact, are sold by a 10-person salesforce. They have also achieved less market penetration.

- To broaden coverage effectively, all competitors list their products in such directories as the ICP Quarterly, Datapro and Auerbach. It is also important to be listed in new product sections of communications-oriented magazines.

- One method to get broad sales coverage quickly would be to have the product distributed by a vendor of a TP monitor or a data base system that does not offer a timesharing system.

C. SYSTEMS SUPPORT

- It is very important to have user-installable software and easy-to-understand documentation. If this is achieved, a vendor does not need systems engineers to support each installation.

- Larger vendors can afford an SE in each sales office because of the number of installations as well as the number of products sold.

- Small vendors can get by with no support in the field if they make the initial investment in testing and documenting the product.

D. A COMPLETE PRODUCT LINE

- A vendor with more than one product gains several advantages.

  - He is perceived as a more stable vendor.

  - He will be able to maximize sales and support resources.

  - He will be less vulnerable to a product announcement by IBM.

- Having only a single product to offer would make it virtually impossible to get started on your own without a large up-front investment.

APPENDIX A: PRODUCT LITERATURE

APPENDIX A: PRODUCT LITERATURE

- This appendix contains vendor-provided documentation on the following products:

  - Tone-3 and Tone-4.

  - Wylbur.

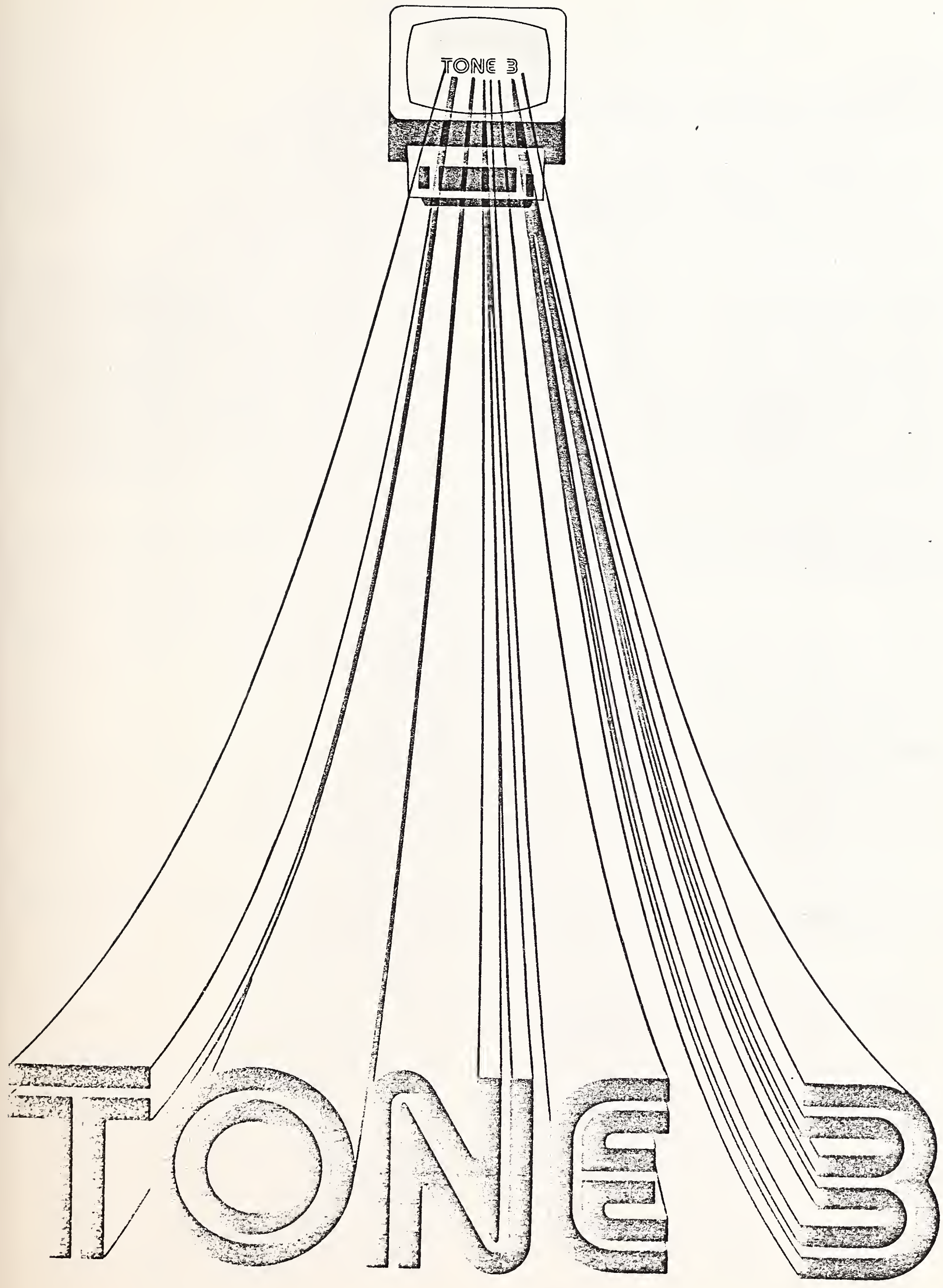
  - Superwylbur.

  - ROSCOE.

  - Interact.

  - IBM SPF.

TONE-3 AND TONE-4


TONE 3
TONE 3

TONE 3

CAPABILITY

TONE 3 provides the capability for OS/VS1 installations to extend the following batch capabilities to the terminal user in a TSO compatible format.

- Edit standard VS1 datasets.
- Submit jobs directly to the JES job queue.
- Review jobs directly from the JES output spool.
- Interactively execute system commands, programs and utilities.
- Inquire system status.

OVERVIEW

The system is designed to meet the following goals:

- Improve the effectiveness of the users computer system.
- Provide a time sharing system facility designed for the VS1 system.
- Provide command compatible with IBM TSO.
- Reduce the overhead associated with TCAM.
- Eliminate the overhead associated with user swapping.
- Minimize the scheduling/operator interaction with the terminal user.

AVAILABILITY: TONE 3 extends to the terminal user the batch capabilities of job submittal and output retrieval. Furthermore, TONE 3 provides for the interactive execution of programs, using the terminal as an input device, and concurrently, an output device.

USEABILITY: These capabilities are provided in a format which is compatible with the IBM time sharing option (TSO); this language has become the standard for IBM computer time sharing users. Significant enhancements to useability have been provided for the users of 3270 type terminals through the TONE 3 3270 support package.

EFFICIENCY: The implementation of most time sharing systems has added significant overhead for the purpose of managing the telecommunications network and the interaction of the multiple online users. The IBM MVT/SVS systems include these components; they are telecommunications access method (TCAM), and user swapping. TONE 3 manages most terminal networks using basic telecommunications access method (BTAM), a simpler and more efficient telecommunications monitor. TONE 3 throughput is enhanced by its use of the multitasking and virtual memory management elements of the VS1 operating system, which eliminate the requirement for user swapping.

TIMELINESS: TONE 3 provides the terminal user with the capability to interact with the computer when it is desireable. The computer resources are immediately allocated to process the data as it is input, and results may be displayed for the user at a time when those results may still impact the process.

PRODUCTIVITY: The computer system is most effective when it is available to the user, easy to use, and delivers the results to the user in a timely manner. TONE 3 manages the computer resources so that the input, processing and output can be performed at the times and at a rate which will allow users to maximize their results

HIGHLIGHTS

- TSO compatible terminal command language.
- TSO compatible command processors.
- Powerful dataset editor
- Interactive program execution.
- Dynamic Allocation.
- Hot internal reader.
- Output using QMGR I/O.
- User swapping eliminated.
- TCAM replaced.
- VTAM support available.
- Standard file structure support.
- Designed for virtual enviornment.
- Powerful command language.
- Enhanced 3270 support package.

TSO COMPATIBLE TERMINAL COMMAND LANGUAGE: The TONE 3 terminal commands are compatible with IBM MVS R3.0 TSO. A terminal user with TSO experience will find that experience applicable to the TONE 3 environment. The recommended manual for terminal users is the OS/VS2 TSO Command Language Reference — VS2 Release 3.0 (GC28-0646). This manual may be supplemented by the TONE 3 user guide.

TSO COMPATIBLE COMMAND PROCESSORS: TONE 3 includes several enhancements to the VS1 operating system. Through these enhancements and the management facilities of TONE 3, most command processors (CP's) written for execution under TSO will operate without modification under the control of the TONE 3 system.

VERSATILE DATASET EDITOR: Dataset editing facilities support all major languages which are in use today. Each of these dataset types are recognized by name, appropriate fields are dedicated to line numbering, convenient tab locations are defined by default, record formats are assumed for new datasets, and appropriate character modifications are automatic (lower case converted to upper case). The following language dataset types are supported:

Cobol;
Fortran;
PL1;
Assembler;
VS Basic;

Other data set types which are supported are:

DATA — for upper case data.
TEXT — for upper and lower case data.
CNTL — for job control streams.

The editor uses a number of the facilities provided by the TONE 3 system. When the user issues the EDIT command, the Editor dynamically allocates the file specified by the user; creates a copy of the specified file in a dynamically allocated work file; and invokes subcommand processors to perform required functions on the work file.

SUPPORTED SUBFUNCTIONS INCLUDE BUT ARE NOT LIMITED TO:

Input	— Places the system in a status to accept data into the work file.
Verify	— Displays at the terminal the line or lines most recently acted upon.
Find	— For examining each record (line) in the work file for a specified sequence of characters.
Change	— Modifies a sequence of characters (character string) in a line or range of lines.
Delete	— Removes one or more records from the work file.
List	— Displays one or more lines of the work file at the terminal.
Move	— Moves one or more records that already exist in the work file.

TONE 3

SUBFUNCTIONS (Cont'd)

Copy	— Copies one or more records that already exist in the work file.
Help	— Obtains the syntax and functions of subcommands.
Submit	— Submits one or more batch jobs for conventional processing. Jobs are entered directly into the JES job queue.
Save	— Retains the edited dataset as a permanent cataloged dataset. The results of the edit may be saved with a new dataset name or replace the dataset originally edited.

INTERACTIVE PROGRAM EXECUTION: Most programs which are executable in batch may be called into execution under the. user's terminal function and either made to interact with the user or simply run to completion with user notification. Normal batch data facilities are available to programs executing interactively; however, interactive programs may receive data directly from the users terminal, and the output from the program may be displayed directly on the terminal.

DYNAMIC ALLOCATION: The dynamic allocation facility allows periodic aquisition and release of resources necessary to simulate batch execution in the foreground. It is as though the execution JCL for this foreground job (the terminal session) had been changed from time to time, during the session, but without the imposition of the initiation/termination overhead.

Dynamic allocation provides access to files which may have been previously created in batch mode, (such as SYS1.LINKLIB), or created in another terminal session. The files may be new files for the user; such as library datasets of JCL, Assembler, Cobol, etc. Files such as SYSIN, SYSOUT, - and SYSPRINT may be directed to the terminal for interactive execution.

Dynamic allocation is also used by TONE 3 to aquire access to terminals. Allocation of the terminals occurs at the time they are started: they are released when they are no longer in use.

HOT INTERNAL READER: Jobs submitted for batch execution are submitted directly to the JES input queues. No intermediate files are used. The hot internal reader (TONERDR) is available to users for job submittal by other batch jobs if desired.

OUTPUT USING QMGR I/O: Upon completion of a submitted job the terminal user can review the spooled output. Through the use of the output command the user may read the SYSMSG dataset and all SYSOUT files directly from the VS1 output queue. The output command interfaces directly with JES using queue manager I/O commands. At any time the user may either:

1. Requeue the output for later review.
2. Requeue the output into a print class destined for hard copy printing.
3. Save the output into an OS sequential or partitioned dataset.
4. Delete all spooled output

USER SWAPPING ELIMINATED: MVT and SVS TSO use a technique called user swapping. The original purpose of user swapping was to provide effective memory management for SYSTEM/360 computers. Whenever a users 'time slice' was satisfied whether or not his function was completed, the users code was written into a swap file, and another users code read in and executed after all other users received an opportunity to execute for their 'time slice' the first users code was retrieved. This user swapping imposed tremendous system overhead.

The virtual memory management design of the VS1 operating system is far more efficient. VS1 memory utilization is based on a least recently used algorithm. This method has proved to be very efficient for interactive processes. TONE 3 uses the VS1 memory management in place of user swapping.

TCAM REPLACED: All IBM versions of TSO use the telecommunications access method (TCAM) for communication with terminals. TCAM management techniques for message handling use significant computer resources, with limited benefits for the time sharing user. TONE 3 utilizes basic telecommunications access method (BTAM). BTAM uses less memory, requires less processing time, and is considerably more reliable.

VTAM SUPPORT AVAILABLE: TONE 3 provides an interface to the virtual telecommunications access method (VTAM). User terminals may interface with multiple TONE regions, and/or other applications through the use of VTAM.

STANDARD FILE SUPPORT: All of the datasets used by TONE 3 have standard OS file structures. TONE 3 users may utilize sequential or partitioned datasets of any record format, logical record length, or block size. Interactive programs can gain the use of previously allocated virtual storage access method (VSAM) datasets. VSAM datasets may not, however, be created through the use of the TONE 3 allocate functions; allocation (creation) of these files must be directed through an interactive execution of the IDCAMS utility.

DESIGNED FOR VIRTUAL ENVIRONMENT: A primary objective in the design of TONE 3 was to effectively utilize the functions of the VS1 operating system. The TONE 3 system is entirely virtual and does no page fixing of its own. The only page fixing is done by IOS for data buffers. TONE 3 code is reentrant, and all code is placed in link pack to minimize the use of both virtual and real storage. TONE 3 may be operated concurrently with the VS1 time slicing and dynamic dispatching algorithms. These can be used in the VS1 system at the selection of the systems programmer to distribute the resources among the various users.

EXTENSIVE COMMAND LANGUAGE: TONE 3 utilizes the TSO command procedure (CLIST) format. These CLISTS are a prearranged executable sequence of TONE 3 commands, subcommands, and control information. CLISTS reside in standard format datasets. A CLIST may be invoked by issuing the EXEC command or the EXEC subcommand of EDIT.

ENHANCED 3270 SUPPORT PACKAGE

- **Full screen reads and writes —**
  Full screen mode reads and writes are supported, utilizing all 1920 characters of the 3277 MOD 02 or compatible device.
- **Windowing —**
  The 3270 screen buffers contain 133 character lines. The windowing function is activated through the use of a program function key (PFK) defined by the user. The interrupt directs the screen of the 3277 to display any selected 80 characters from the possible 133. The defaults are 1-80 and alternately 54-133. Using this windowing it is convenient to review data, such as some SYSOUT, which is wider than the physical terminal screen.
- **Physical Tab —**
  Through the PFK functions the user can define a physical tab. Execution of this function will cause the cursor to skip along the input line filling in the skipped area with blanks.

- **Logical Tab —**
  The user may define a character to be used as a logical tab character. The 3270 support will replace the logical tab character in the input line so that the receiver of the data line will recognize the real tab (x'15') character.
- **Programmable PA/PF Keys —**
  TONE 3 supports programmable PA/PF keys. The user can retrieve the PA/PF key definitions and modify any of the 15 possible keys for special functions, or to input data to the system. Any PF function can be made to be interactive, the data line is inserted in the input area of the 3277 and the cursor is placed in the line as directed by the user. The modified definitions may be saved for later recall by member name. The functions are executed when the PA/PF keys are depressed, or when the PA/PF key simulation is executed.
- **PA/PF Key Simulation —**
  For those terminals without a full set of PA or PF keys, TONE 3 provides a PA/PF key simulation method. This function enables the terminal user to indicate the PA/PF function desired, and the appropriate data, if any, to be used in the PA/PF function.
- **Instant Interrupt —**
  Most TONE 3 functions are instantly interruptable from the 3277. This is in sharp contrast to TCAM which required the users to specify a time interval, after which the system must interrogate the terminal for permission to continue. The 3277 is always in ready status when using TONE 3.
- **Line Retrieval —**
  The terminal user may, using the line retrieval functions, retrieve a previously displayed line or part of a line. This is done by moving the cursor to the beginning of the data desired, and depressing the enter key.
- **Screen Format —**
  The TONE 3 system displays at all times, (except when in full screen mode), a format which conveniently contains: the time of day; the time since logon; the total CPU time used; the identity of the current user; and the command in execution.

COMPONENTS

The TONE 3 system is comprised of the following major components.

- **TONE 3 VS1 enhancements**
  - Dynamic allocation
  - CPU timing for subtasks
  - Memory Subpool Management
  - CVT extention
  - TSO services
- **TONE 3 partition code**
  - TONE 3 region controller
  - TONE 3/TSO initiator
  - All BTAM console handlers
  - Console/Program interface
  - MVS (VS2 R3.0) TSO to function under VS1
  - Edit
  - Allocate
  - Free
  - Attribute
  - Call
  - Submit
  - Output
  - Logon/Logoff
  - List/ds/cat

SYSTEM CONFIGURATION

- **Virtual Storage**
  The virtual storage requirement is dependent upon the number of active TONE 3 partitions and the modules selected for link pack. Approximately one megabyte in link pack and a one megabyte partition for each twelve started terminals is necessary. Additional terminals require only partition space.
- **Real Storage**
  The real storage utilization is dependent upon the activity of the system, the relative priority of the TONE 3 partitions, and the nature and frequency of user commands. It is suggested that the user monitor memory utilization during the trial period to determine the nature of this requirement.
- **Partitions**
  TONE 3 supports multiple users in all TONE 3 partitions. The number of users is limited by the VS1 limitation of 255 "DD" statements per partition. Approximately ten to twelve started terminals are possible for each TONE 3 partition.
- **DASD Storage**
  The TONE 3 code will require approximately one megabyte of online DASD storage. Approximately ten to twelve megabytes may be offline at times other than installation.
- **Central Processing Units**
  System/370 Models 135 and up, 433X, 434X, 303X, and other announced IBM or compatible machines supporting VS1 operating systems.
- **Direct Access Storage**
  2314 direct access storage facility, (models 1, A and B) and 2844 auxiliary storage control;
  2319 disk storage, (models A and B);
  3330 series disk storage, all models;
  3340 direct access storage facility, all models;
  3350 direct access facility, all models;
  2305 fixed head storage facility, models 1 and 2.
- **Terminals**
  Master console;
  3270 Model 2;
  ASCII devices;
  2740, models 1 and 2 (station control and checking are necessary);
  2741
  3278 - all models including 132 character model 5

EDUCATION

The dates, times and locations of TONE 3 seminars and workshops will be selected to best suit the needs of TSC customers.

TECHNICAL SUPPORT

Programming support, enhancements, and technical assistance are included for the term of the contract. Thereafter support will be available from TSC at a nominal monthly charge.

FREE TRIAL PERIOD

The user will have thirty days to evaluate the use of the TONE 3 system. If at any time during this period the user decides not to continue, they may return the software without further obligation.

INSTALLATION AND TRAINING

Installation by Tone Software Corporation personnel is recommended TSC's installation charge includes a one day class in the use of the TONE 3 facilities.

TONE
Software Corp.

(714) 991-9460
1735 SO. BROOKHURST
ANAHEIM, CA 92804
TELEX: 181592

TONE 4 MVS

TONE 4

CAPABILITY

TONE 4 enables MVS installations to extend the following capabilities to the terminal user.

- Run multiple users in a single address space.
- 133 character logical screen with left and right scrolling.
- User defined program function keys.
- Instant interrupt.

OVERVIEW

The system is designed to meet the following goals:

- To improve the effectiveness of the users computer system.
- To make this capability compatible with IBM TSO.
- To eliminate the overhead associated with TCAM.
- To eliminate the overhead associated with user swapping.
- To minimize the scheduling/operator interaction with the terminal user.
- To minimize the effects of the interactive time sharing facilities on computer resources.

USEABILITY: TONE 4 capabilities are provided in a format which is identical to the IBM time sharing option (TSO); this language has become the standard for IBM computer time sharing users. Significant enhancements to useability have been provided for the users of 3270 type terminals through the TONE 4 3270 support package.

EFFICIENCY: The implementation of most time sharing systems has added significant overhead for the purpose of managing the telecommunications network and the interaction of multiple online users. The IBM MVT/SVS/MVS systems include these components; they are telecommunications access method (TCAM), and user swapping. TONE 4 manages most terminal networks using basic telecommunications access method (BTAM), a simpler and more efficient telecommunications monitor. TONE 4 throughput is enhanced by its use of MVS multitasking and virtual memory management, which eliminate the requirement for user swapping.

EFFECTIVENESS: The computer system is most effective when it is available to the user, easy to use, and delivers the results to the user in a timely manner. TONE 4 manages the computer resources so that the input, processing and output can be performed at a rate which will allow users to maximize their results.

HIGHLIGHTS

- User swapping eliminated.
- TCAM replaced.
- Enhanced 3270 support package.
- VTAM support available.
- Designed for virtual environment.
- Standard file structure support.
- Versatile TSO Data Set Editor.
- TSO Interactive Program Execution.
- Output using QMGR I/O.
- TSO terminal command language.
- TSO command processors.
- Full screen edit option availability.

USER SWAPPING ELIMINATED: MVT, SVS and MVS TSO use a technique called user swapping. The original purpose of user swapping was to provide effective memory management for SYSTEM/360 computers. Whenever a user's 'time slice' was satisfied, whether or not his function was completed, the user's code was written into a swap file, and another user's code read in and executed; after all other users received an opportunity to execute for their 'time slice' the first users code was retrieved. This user swapping imposed tremendous system overhead.

The virtual memory management design of the MVS operating system is far more efficient. MVS memory utilization is based on a least recently used algorithm. This method has proved to be very efficient for interactive processes. TONE 4 uses the MVS memory management in place of user swapping.

TCAM REPLACED: All IBM versions of TSO use the telecommunications access method (TCAM) for communication with terminals. TCAM management techniques for message handling use significant computer resources, with limited benefits for the time sharing user. TONE 4 utilizes basic telecommunications access method (BTAM). BTAM uses less memory, requires less processing time, and is considerably more reliable.

ENHANCED 3270 SUPPORT PACKAGE

- **Full screen reads and writes —**
  Full screen mode reads and writes are supported, utilizing all 1920 characters of the 3277 MOD 02 or compatible device.
- **Windowing —**
  The 3270 screen buffers contain 133 characters per line. The windowing function is activated through the use of a program function key (PFK) defined by the user. The interrupt directs the screen of the 3277 to display any selected 80 characters from the possible 133. The defaults are 1-80 and alternately 54-133. Using this windowing it is convenient to review data, such as SYSOUT, which is wider than the physical terminal screen.
- **Physical Tab —**
  Through the PFK functions the user can define a physical tab. Execution of this function will cause the cursor to skip along the input line filling in the skipped area with blanks.
- **Logical Tab —**
  The user may define a character to be used as a logical tab character. The 3270 support will replace the logical tab character in the input line so that the receiver of the data line will recognize the real tab (x'15') character.
- **Programmable PA/PF Keys —**
  TONE 4 supports programmable PA/PF keys. The user can retrieve the PA/PF key definitions and modify any of the 15 possible keys to effectuate special functions, or to input data to the system. Any PF function can be made to be interactive; the data line is inserted in the input area of the 3277 and the cursor is placed in the line as directed by the user. The modified definitions may be saved for later recall by member name. The functions are executed when the PA/PF keys are depressed, or when the PA/PF key simulation is executed.
- **PA/PF Key Simulation —**
  For those terminals without a full set of PA or PF keys, TONE 4 provides a PA/PF key simulation method. This function enables the terminal user to indicate the PA/PF function desired, and the appropriate data, if any, to be used in the PA/PF function.
- **Instant Interrupt —**
  Most TONE 4 functions are instantly interruptable from the 3277. This is in sharp contrast to TCAM which required the users to specify a time interval, after which the system must interrogate the terminal for permission to continue. The 3277 is always in ready status when using TONE 4.

TONE 4

- **Line Retrieval —**
The terminal user may, through the facility of the line retrieval functions, retrieve, and place in the input area, a previously displayed line or part of a line. This is done by moving the cursor to the beginning of the data desired, and depressing the enter key.
- **Screen Format —**
The TONE 4 system displays at all times (except when in full screen mode), a format which conveniently contains: the time of day; the time since logon; the total CPU time used; the identity of the current user; and the primary command name.

AM SUPPORT AVAILABLE: TONE 4 provides an interface the virtual telecommunications access method (VTAM). ser terminals may interface with multiple TONE regions, d/or other applications through the use of VTAM.

ESIGNED FOR VIRTUAL ENVIRONMENT: A primary objec- e in the design of TONE 4 was to effectively utilize the nctions of the MVS operating system. The TONE 4 system entirely virtual and does no page fixing of its own. The only ige fixing is done by IOS for data buffers. TONE 4 code is re- itrant; code placed in link pack minimizes the use of both tual and real storage.

TANDARD FILE STRUCTURE SUPPORT: All of the datasets ed by TONE 4 have standard OS file structures. TONE 4 ers may utilize sequential or partitioned datasets of any cord format, logical record length, or block size. Interactive ograms can gain the use of previously allocated virtual stor- ie access method (VSAM) datasets. VSAM datasets may t, however, be created through the use of the TONE 4 allo- te functions; allocation (creation) of these files must be rected through an interactive execution of the IDCAMS ility.

ERSATILE TSO DATASET EDITOR: Dataset editing facilities pport all major languages which are in use today. Each of ese dataset types are recognized by name, appropriate elds are dedicated to line numbering, convenient tab cations are defined by default, record formats are assumed r new datasets, and appropriate character modifications are tomatic (lower case converted to upper case). The following nguage dataset types are supported:

COBOL;

FORTRAN;

PL1;

ASM;

VS BASIC.

her data set types which are supported are:

ATA — for upper case data.

EXT — for upper and lower case data.

NTL — for job control streams.

JPPORTED SUBFUNCTIONS INCLUDE, BUT ARE NOT MITED TO:

put	—	Places the system in a status to accept data into the work file.
Verify	—	Displays at the terminal the line or lines most recently acted upon.
Find	—	For examining each record (line) in the work file for a specified sequence of characters.
Change	—	Modifies a sequence of characters (character string) in a line or range of lines.
Delete	—	Removes one or more records from the work file.
List	—	Displays one or more lines of the work file at the terminal.
Move	—	Moves one or more records that already exist in the work file.
Copy	—	Copies one or more records that already exist in the work file.
Help	—	Obtains the syntax and functions of subcommands.
Submit	—	Submits one or more batch jobs for conventional processing. Jobs are entered directly into the JES job queue.
Save	—	Retains the edited dataset as a permanent cataloged dataset. The results of the edit may be saved with a new dataset name or replaced with the dataset originally edited.

TSO INTERACTIVE PROGRAM EXECUTION: Most programs which are executable in batch may be called into execution under the user's terminal function and either made to interact with the user or simply run to completion with user notification. Normal batch data facilities are available to programs executing interactively; interactive programs however, may receive data directly from the users terminal, and the output from the program may be displayed directly on the terminal.

TSO OUTPUT USING QMGR I/O: Upon completion of a submitted job the terminal user can review the spooled output. Through the use of the output command the user may read the SYSMSG dataset and all SYSOUT files directly from the MVS output queue. The output command interfaces directly with JES using queue manager I/O commands. At any time the user may either:

1. Requeue the output for later review.
2. Requeue the output into a print class destined for hard copy printing.
3. Save the output into an OS sequential or partitioned dataset.
4. Delete all spooled output.

TSO TERMINAL COMMAND LANGUAGE: The TONE 4 terminal commands are identical with IBM MVS R3.7 TSO. A terminal user with TSO experience will find that experience applicable to the TONE 4 environment. The recommended manual for terminal users is the OS/VS2 TSO Command Language Reference — VS2 Release 3.7 (GC28-0646). This manual may be supplemented by the TONE 4 user guide.

TSO COMMAND PROCESSORS: TONE 4 supports TSO command processors. Most command processors (CP's) written for execution under TSO will operate without modification under the control of the TONE 4 system.

FULL SCREEN EDIT OPTION AVAILABILITY: Separate price option. (Specification available upon request.)

SYSTEM CONFIGURATION

- Virtual Storage
  The virtual storage requirement is dependent upon the number of active TONE 4 partitions and the modules selected for link pack (approximately one megabyte in link pack and one to two megabytes in each address space). Additional terminals require only address space.

- Real Storage
  The real storage utilization is dependent upon the activity of the system, the relative priority of the TONE 4 address spaces, and the nature and frequency of user commands. It is suggested that the user monitor memory utilization during the trial period to determine the nature of this requirement.

- Address Spaces
  TONE 4 supports multiple users in each MVS address space. The number of users is theoretically unlimited. Approximately 15 to 25 started terminals are recommended for each TONE 4 partition.

- DASD Storage
  The TONE 4 code will require approximately one megabyte of online DASD storage. Approximately ten to twelve megabytes may be offline at times other than installation.

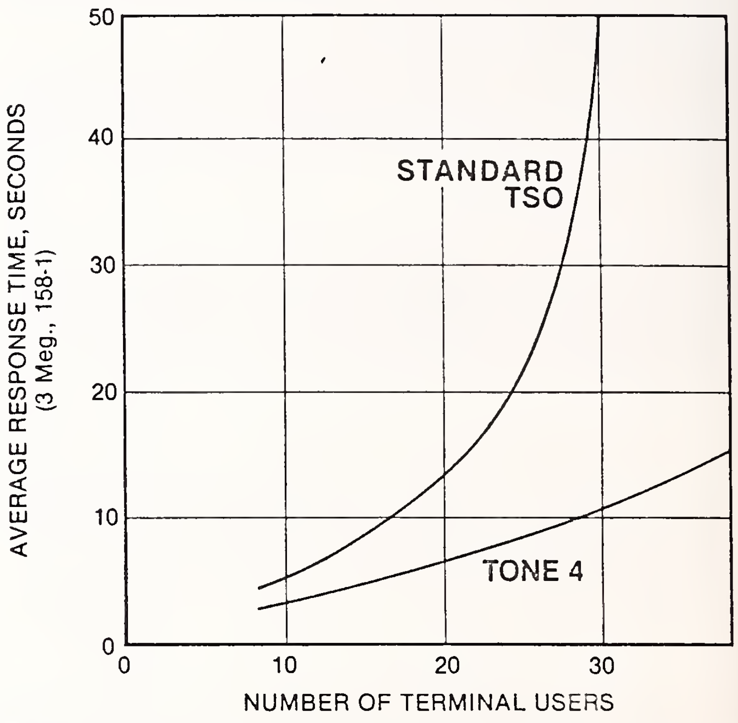
- Central Processing Units
  IBM System/370 Models 158, (models 1 and 3), 165 II, 168 (models 1 and 3); 3032, 3033, or equivalents.

- Direct Access Storage
  IBM 3330 series disk storage, all models;
  IBM 3340 direct access storage facility, all models;
  IBM 3350 direct access facility, all models;
  IBM 2305 fixed head storage facility, models 1 and 2.
  Equivalents of any of the above storage devices.

- Terminals
  Master console;
  3270 model 2;
  ASCII devices;
  2740, models 1 and 2 (station control and checking are necessary);
  2741.
  Equivalents of any of the above terminal devices.

PERFORMANCE

Data varies depending upon CPU, and type of job involved. The resultant graphs depict typical response times for standard TSO and TONE 4.



EDUCATION

The dates, times and locations of TONE 4 seminars and workshops will be selected to best suit the needs of TSC customers.

TECHNICAL SUPPORT

Programming support, enhancements, and technical assistance are included for the term of the contract. Thereafter support will be available from TSC at a nominal monthly charge.

FREE TRIAL PERIOD

The user will have thirty days to evaluate the use of the TONE 4 system. If at any time during this period the user decides not to continue, they may return the software without further obligation.

INSTALLATION AND TRAINING

Installation by Tone Software Corporation personnel is recommended. TSC's installation includes one day on site training in the use of TONE 4 capabilities.

TONE Software Corp.

(714) 991-9460
1124 N. Gilbert Anaheim, Ca. 92801

DASD3336 SECURITY—EVALUATION—MONITORING

CAPABILITY

DASD3336 is designed to provide complete security, analysis and history of 3336 Mod 1 and 3336 Mod 11 disk packs. The following functions are supported:

1. Overwrites all tracks, whether active or currently assigned alternates.
2. Lists all tracks currently assigned alternates.
3. Attempts reuse of an original track if currently assigned an alternate.
4. Completely reformats a pack, rewriting all HA/R0's.

OVERVIEW

The program was designed to meet the following goals:

1. To provide the user with certainty that all high-security data on the pack has been destroyed.
2. To enable the purchaser of 3336 or equivalent packs to fairly and impartially evaluate all data areas of each pack submitted for aquisition.
3. To provide for ongoing monitoring of pack performance.
4. To provide the user with the capability to improve performance by reusing a primary track now assigned an alternate, if the primary track is now able to accept data.
5. To reformat a currently unusable volume.

Operator Assistance

During analysis operations DASD3336 displays the results of each function on the console to enable the operator to monitor the progress of the analysis.

List Function

The list option allows the listing of alternate tracks assigned to a pack. This option may be run online to any 3330 Mod 1 or Mod 11 while the pack is being used for normal operation.

Retrieve Function

The retrieve option attempts to reactivate a primary track which is currently assigned an alternate. DASD3336 reads the data on the alternate and attempts to rewrite it on the primary track. If this is successful, the primary track is marked as active. This option must be run with the drive offline to OS/VS1.

Format Function

The format option provides the analysis of each track of a 3336 pack. This analysis checks each bit cell on primary and alternate tracks. A temporary VTOC is written on tracks 1 thru 5 with a label specified by the user. This option must be run with the drive offline to OS/VS1. This function provides complete destruction of all data currently on the volume.

PERFORMANCE

The retrieval and format functions have a 'passes' parameter. Through the use of this parameter the user may specify the extent of the analysis. Each pass requires approximately fifteen (15) minutes for a Mod 1 pack and thirty (30) minutes for a Mod 11 pack.

HARDWARE/SOFTWARE CONFIGURATION

DASD3336 requires a 370 CPU with a minimum of one available 3330 type drive. Appropriate 370 CPU's include: 370/135, 370/138, 370/145, 370/148, 370/155 II, 370/158 Mod 1 and Mod 3, 370/165 II, 370/168, 3031, 3031-MP, 3032, 3032-MP, 3033, 3033-MP. DASD3336 is currently supported under OS/VS1 release 6.0 through the most current release of OS/VS1.

Technical Support

Programming support, enhancements, and technical assistance are included for the initial term of the contract. Thereafter, support will be available from TSC at a nominal monthly charge.

Ordering Information

DASD3336 with its associated documentation is currently available. Contact TSC to order the package.

TLIB INTERACTIVE LIBRARIAN COMMAND PROCESSOR

OVERVIEW

TLIB, Interactive LIBRARIAN Command Processor, provides the user with the ability to fully utilize the resources of LIBRARIAN when operating in a TONE or TSO environment. The user may store, retrieve, copy or delete datasets interactively using a free form command structure. System and programmer efficiency are improved through:

A reduced requirement for disk space by storing modules not currently needed on the LIBRARIAN master file in a compressed format.

Powerful on-line editing facilities available through TONE and TSO that provide several functions not available with the LIBRARIAN editor.

Extensive prompting on all commands which greatly reduces programmer training and simplifies the use of all LIBRARIAN functions.

COMMAND FUNCTIONS

TLIB has seven basic commands which provide a complete interface between the LIBRARIAN and TONE/TSO, enabling the user to have complete and flexible access to all LIBRARIAN functions. These commands include:

LIBGET enables a user to retrieve information from the LIBRARIAN master file. The selected module is assigned to a dynamically allocated standard format VS data set in expanded form and immediately available for review or subsequent editing using the powerful TONE/TSO edit capabilities.

LIBSAVE enables terminal users to add or replace modules on the LIBRARIAN master file. During an ADD operation, prompting is provided for description and language parameters.

LIBEDIT allows the terminal user to manipulate both the column and sequence number positioning of data stored in a LIBRARIAN module. Accuracy is easily examined using the LIBLOOK function.

LIBFILL enables the operator to "gang punch" data into specified LIBRARIAN modules.

LIBDLM deletes a module from the master file.

LIBRENME allows the user to rename a specified LIBRARIAN module.

LIBLOOK is a utility command cluster providing the terminal user with extensive access to stored materials. Five options include:

LIBLOOK INDEX displays a full index of all modules available on the Librarian master file.

LIBLOOK INDEX(XXXXX) displays a complete index of modules by programmer ID.

LIBLOOK LISTH displays records associated with the desired module as well as control information specifically pertaining to that module.

LIBLOOK LIST allows the user to access the control information available with the LIBLOOK LISTH command and additionally provides a complete list of the module itself.

LIBLOOK PUNCH command provides the user with access to data sets by using the utility or punch option. It allows modules containing INC's to be retrieved in nonexpanded form.

PROGRAMMING SYSTEMS

TLIB is supported under current releases of TONE 3, TONE 4 MVS and MVS TSO.

TECHNICAL SUPPORT

Programming support, enhancements, and technical assistance are provided for the term of the initial contract. Thereafter, support will be available from Tone Software Corporation at a nominal charge.

INSTALLATION

While it is recommended that the customer install TLIB, installation is available from Tone Software Corporation on a time and expenses basis.

TRIAL PERIOD

Tone Software Corporation offers the customer 30 days to evaluate TLIB software and associated documentation free of charge and without further obligation.

TONE

Tone Software Corporation
[illegible]
Anaheim, California 92801
Phone [illegible] 991-4400
Telex [illegible] 592

PHASOR VS1 DYNAMIC ALLOCATION OF SYSOUT

CAPABILITY

PHASOR provides an OS/VS1 user with the ability to dynamically allocate and release SYSOUT datasets for early printing. The function is similar to the DYNALOC feature offered in MVS with a number of extensions.

OVERVIEW

Long-Running Tasks

Long-running tasks such as TP monitors need not be terminated to print status reports or dumps. These datasets may be created dynamically and a facility added to the program to release the SYSOUT on command.

Exception Reports

Long-running application programs may release exception reports as soon as they are available. The exception reports may be examined while the application itself is still processing.

Dynamic Class and Destination

User programs with PHASOR may specify, at execution time, for dynamic SYSOUT datasets:

output class	jobname
destination user ID	output writer
number of copies	accounting info
output priority	form type

Spool Efficiency

VS/1 spool space allocated for a job is not freed until all SYSOUT generated by the job has been printed. This characteristic may create severe problems in a RES environment. Suppose, for example, the user has a job which creates 100,000 lines for DEST 'A' and 10 lines for DEST 'B'. Subsequently, user 'A' prints the 100,000 lines; until user 'B' prints the 10 lines, all of the spool space for the job will remain allocated. PHASOR dynamic SYSOUT datasets are viewed by OS/VS1 as independent jobs. Spool space for each dynamic SYSOUT dataset is released as soon as printing is complete for that dataset.

INSTALLATION

PHASOR is added to the operating system as a type three or type four SVC. No other modifications to the operating system are required.

IMPLEMENTATION

The JCL/program changes which are required to make use of PHASOR are:

1. The DD SYSOUT = X keyword for SYSOUT datasets, which are being changed to PHASOR dynamic SYSOUT datasets, must be replaced with DD DYNAM or DD DUMMY parameters.

2. Prior to opening a PHASOR dynamic SYSOUT dataset, the user must issue a call to the PHASOR SVC.

3. To release the PHASOR dynamic SYSOUT dataset, the user must issue another call to the PHASOR SVC. The user may, at this time, specify CLASS, DEST, COPIES, etc. for this dataset. The dataset need not be closed to release it for processing; however, it should be closed the last time this call is issued for any given dataset.

4. If for any reason the user wishes to revert to conventional SYSOUT processing, the user must replace the DYNAM or DUMMY parameter with the standard SYSOUT = X keyword. No program modifications would be required at that time.

PROGRAMMING SYSTEMS

PHASOR is currently supported under OS/VS1 release 6.0 through the most current release.

TECHNICAL SUPPORT

Programming support, enhancements and technical assistance are included in the lease price. Under a Fully Paid License, support is included at no charge for the first year, and available for a nominal fee thereafter.

TRIAL PERIOD

TONE SOFTWARE CORPORATION offers the customer a 30-day trial period to evaluate FSM. If at any time during this trial period the user decides not to continue the evaluation, the system and documentation may be returned without obligation.

INTERSYSTEM SPOOL PROCESSOR

The Intersystem Spool Processor (ISP) is designed to provide total flexibility for multiple OS/VS1 systems in the processing of input and output spools. It provides for the complete routing of input jobs as well as created output between any processors sharing in the ISP environment.

The user can now pool printer and/or reader-punch resources on the processor that will give the most efficient spool processing. This capability can significantly enhance the utilization of these critical and expensive resources. Installations using ISP can determine the need for redundancy of these system components, based upon each component's contribution to system reliability and effectiveness.

OPERATING EFFICIENCY

Transmission of output data to subsequent CPU's is implemented using shared DASD with redundancy data compression. This offers significant space saving when compared to the JES spool.

OPERATOR ASSISTANCE

During processing, ISP displays the number of print lines in each data-set. This capability will facilitate the scheduling of extensive output.

CRITICAL SPOOL

Each CPU can route to any CPU in the ISP environment. Using this function a critical spool situation may be resolved by routing output data into the same system thus relieving the overload.

COMPRESSION

ISP uses a data compression algorithm to compress redundant data in any output lines to be transferred to another CPU. This procedure will reduce the space required for the output data by approximately thirty-eight percent.

HARDWARE SCHEDULING

Pooling of I/O on one CPU will allow JES to optimize the use of the available peripherals. Overall turnaround time may be substantially reduced.

HARDWARE CONFIGURATION

Each processor no longer requires a dedicated card reader and printer. If the volume of cards read or lines printed is not sufficient to warrant the current quantity, one or more of the unnecessary units may be eliminated. Since the units may now be consolidated on one processor substantial savings in peripheral control units are also possible.

PROGRAMMING SYSTEMS

The system is designed to operate under release 6.0 and all current releases of the IBM OS/VS1 Operating System.

SYSTEM CONFIGURATION

Shared direct access space is required for the transmission of the reader and writer data (approximately 250 tracks for each 1000 pages of spooled data).

The main storage requirement of the system is dependent upon the function of the CPU. Approximately 8192 bytes of virtual storage are required for each system sending output. Approximately 2048 additional bytes of virtual storage are required for the CPU controlling the unit record peripherals.

The operating speed is dependent upon the CPU and Disk devices used. Representative speed for a configuration using a 370/158, and 3330-1 disks is approximately 250 lines per second.

ISP can utilize the following devices provided the necessary support is available in the level of VS-1 being used:

CENTRAL PROCESSING UNITS

370/135, 370/138, 370/145, 370/148, 370/155-II, 370/158, 370/165-II, 370/168, 3031, 3032, 3033

DISK UNITS

2314, 3330, 3330-1, 3340, 3350

UNIT RECORD DEVICES

2540, 2501, 1403-7, 1403-N1, 1443, 3211, 3800

NUMBER OF CPU'S

Concurrently supportable—2-36

TONE 3 — 328X SUPPORT

CAPABILITY

TSC — 328X enables TONE users to logically connect 3270 type display stations to 328X type printer devices, enabling the terminal user to cause specified data to be printed on the connected hard-copy device.

OVERVIEW

- Encourages effective testing as users can now have complete hard copy records of test sessions.
- Selected copies of assembly or compile errors can be printed without printing of entire listings.
- User CP's can directly address both the terminal and printer with independently formatted data.
- Datasets and output listings can be printed at the terminal location to reduce turnaround time.
- Facilitates routing of data between various remote locations.

IMPLEMENTATION

Each printer is assigned in the TONE.LTERM dataset to a logical group. To connect a printer, the terminal user specifies a group name. TONE will select, from the group specified, an available device and effect the connection. If all of the started hardcopy devices in the group specified are already connected, the terminal user is notified that no hard-copy device is available.

When a hard-copy device is connected to a terminal, the mode and LTERM name are displayed on the mode line (the second to last line on a 3270 display station) in the form:

```
COMMAND — HC = PRTI — (STD MODE,LTERM PRTI)
COMMAND — PP = PRTI — (PGPT MODE,LTERM PRTI)
```

The terminal user may cause the printer to:

(1) (HC) Connect the printer, skip the printer to top of form, and begin following the terminal.

(2) (HT) Connect the printer, skip the printer to top of form, print the data currently on the screen, and begin following the terminal.

(3) (TOF) Skip to top of form.

(4) (PGPT) Connect the printer, skip the printer to top of form, enter page print mode. The printer will print only when directed.

(5) (CP) Enter continuous print mode. The printer will print each line as it is directed to the terminal.

(6) ($$PGPT) Print the data currently displayed on the screen.

(7) ($$POFF) Disconnect the printer, freeing it for use by another terminal. The printer is disconnected at Logoff by default.

The terminal user may cause the 3270 type display station to:

(1) ($$IEOP) Ignore the normal end-of-page conditions. The printer will continue to write data without intervention by the terminal user.

(2) ($$AEOP) Acknowledge the normal end-of-page conditions. The printer will be directed to write lines of data until a full screen of data is received by the 3270 display station. It will be refreshed, showing the already hard-copied lines of data, and will require "enter" or "clear" to continue. Depressing the "$$ATTN" PA or PF key will cause an interruption to the active CP and return control to the terminal user simultaneously interrupting the printing of more data. This is the normal mode for the terminal and it is reset to (AEOP) at user Logoff.

Printers which may be connected to the TONE system are defined in the TONE.LTERM dataset. The member name is the name started to activate the printer. At the time the LTERM is started, TONE will write a TONE logo on the printer. When the printing of the logo is complete, the printer is available to any terminal attempting to acquire hard copy via the HC or HT commands.

PROGRAMMING SYSTEMS

328X is currently supported under the current releases of TONE 3 and TONE 4 MVS.

TECHNICAL SUPPORT

Programming support, enhancements and technical assistance are included for the initial term of the contract. Thereafter, support will be available from TSC at a nominal monthly charge.

ORDERING INFORMATION

328X with its associated documentation is currently available. Contact TSC to order 328X.

INSTALLATION

Installation is available from TONE SOFTWARE CORP. Installation charges are based on time and expenses.

TRIAL PERIOD

The user will have thirty days to evaluate 328X in his environment. If at any time during this period the user decides not to continue, he may return the package and all associated documentation without further obligation.

FSO FULL SCREEN OUTPUT

OVERVIEW

FSO, Full Screen Output, provides the 3270 display terminal user with a set of extremely powerful and flexible commands to conveniently review spooled input and output as well as determine the status of all jobs in the system. With FSO the users may look at input as well as output datasets, search for character strings, set note points for later direct reference, page backward or forward, delete the spooled output, requeue it for later review, or print it on the system printer. The output commands act directly on the JES queue and no intermediate file is necessary.

In addition to the powerful output facilities, FSO provides commands which perform the functions of IBM's IEHPROGM utility interactively. FSO users may inquire into the status of input and output queues, display jobs currently being processed, display the status of a partition or address space including I/O counts by DDNAME, paging counts, CPU usage and virtual storage usage.

A complete set of commands is also included to aid the systems programmer. These include the ability to display and alter virtual storage by offset, to locate modules and their associated pointers in the link pack, job pack, or resident SVC area and the ability to display formatted UCB's and TCB's. An absolute cancel facility is also provided.

SPOOLED INPUT/OUTPUT REVIEW

FSO allows the user to easily obtain and manipulate input/output directly from the spool without JCL modifications. Once the data has been retrieved the user may:

- Scroll vertically or horizontally
- Search for a specified character string
- Page forward or backward
- Set reference points for subsequent return
- Delete or requeue for subsequent processing

STATUS COMMANDS

FSO Status Commands permit the user to:

- Display active jobs
- Display jobnames on input or output queues
- Display detailed information about an active partition or job
- Display jobqueue status
- Display outstanding requests

DATA SET UTILITY FUNCTIONS

Through the use of the FSO Data Set Utility Functions the user may perform most functions of the IBM IEHPROGM utility online. Included among these functions is the ability to:

- Connect or delete a high level index
- Create or delete generation data group indicies
- Catalog or uncatalog a dataset
- List the catalog
- Rename or scratch datasets
- Determine the free space on a volume
- List Format 1 DSCB information

ALTER/DISPLAY FUNCTIONS

This set of flexible commands provides the systems programmer with the following capabilities:

- Easily display or alter virtual storage
- Locate modules in LPA or JPAQ
- Locate resident SVC's
- Display TCB's (VS1 only)
- Display UCB's
- Express cancel a TCB (VS1 only)

These commands may be restricted to specified individuals through easily programmed exits.

PARTIAL COMMAND LISTING

The following brief listing is a representative sample of the more than 70 commands currently available with FSO.

DELETE	- Delete job from system
FIND	- Locate specific string
INDEX	- Display SYSOUT index
LIST	- List selected portions to SYSOUT
INPUT	- Retrieve from Input Queue
OUTPUT	- Retrieve from Output Queue
SAVE	- Requeue the current job
DA	- Display active jobs in the system
DN	- Display all jobs on all Queues
DP	- Display an active partition
DR	- Display outstanding requests
STATUS	- Determine the status of a job
DV	- Display virtual storage

PROGRAMMING SYSTEMS

FSO is supported under all current releases of OS/VS1 and MVS and functions under various teleprocessing monitors including TONE 3 and TONE 4 (TONE SOFTWARE CORP.), Roscoe (Applied Data Research, Inc.), TSO and CICS/VS (IBM Corp.) and Shadow (Altergo Software Inc.). FSO also functions in batch or from the operator's console. The display stations supported include any local or remote 3270 or compatible device ranging from 12 to 43 display lines and 80 or 132 columns.

TECHNICAL SUPPORT

Programming support enhancements and technical assistance are included in the lease price. Under a Fully Paid License, support is available at no charge for the first year and for a nominal fee thereafter.

TRIAL PERIOD

TONE SOFTWARE CORPORATION offers the customer a 30-day trial period to evaluate FSO. If at any time during this trial period the user decides not to continue the evaluation, the system and documentation may be returned without obligation.

TONE

FSM FULL SCREEN MODE

OVERVIEW

The Full Screen Mode (FSM) of data editing provides users with the full capabilities of 3270 protocol. The terminal user may, through the 3270, define the field to be edited, position the cursor anywhere on the screen and make the desired edits. FSM has the ability to process the entire screen of new or updated data regardless of screen size. The term 3270 refers to a class of devices including local and remote 3275, 3277, 3278, and 3279 terminals with screen sizes of 24x80, 32x80, 43x80, and 27x132 characters. FSM greatly enhances productivity through:

- Full Screen processing
- Enhanced terminal useability
- Simplified line commands
- Hexadecimal editing
- Extended NONUM support
- New EDIT subcommands

FULL SCREEN PROCESSING: FSM reads and processes an entire screen of new and/or modified data in one service cycle. All processing required for the screen is performed before the display is updated. This is done for all standard screen sizes and results in a significant reduction in overhead for the transmission of data.

ENHANCED TERMINAL USEABILITY: The user is provided with:

Logical Tabs - The user may enter data using the designated character to represent a tab. When data is processed, blanks are inserted to align the data as directed.

Screen Format - The format provides a command line, descriptor area, tab templates, column templates, and a data area, variable in size depending upon the number of lines displayable on the screen.

3270 Features - Within the data area the user may overtype existing data, and use character insertion and deletion to move or align new and/or existing data.

SIMPLIFIED LINE COMMANDS: FSM provides the capability to move, copy or delete single or contiguous blocks of lines simply with the use of one or two character commands. This eliminates the need for entering line numbers and speeds the editing process.

HEXADECIMAL EDIT SUPPORT: FSM provides the ability to create or modify data fields in HEX. The HEX editing feature provides a powerful tool in the development or correction of data. It allows the user to display fields that are hexadecimal or packed decimal and to modify the value of these fields. All data is displayed in 2 character byte representation. Subcommands exist to display only hexadecimal representation or, two lines can be provided, one containing hexadecimal and the other in character format.

EXTENDED NONUM SUPPORT: FSM allows the user to update NONUM datasets as if they were numbered. It can display an edit generated line number on the left side of the line. Initially, these line numbers are incremented by 100, but line insertions may make the increments smaller. This facility of FSM allows the user to execute standard editor line number commands just as if the datasets contained line numbers.

ADDITIONAL EDIT SUBCOMMANDS:

ASIS -	Set the dataset into ASIS mode.
CAPS -	Set the dataset into CAPS mode.
CHAR -	Display the data in the line display in character mode.
COL (n m) -	Change the columns displayed to "n" for "m" columns.
COL C -	Move the display to the column in which the cursor is positioned.
COL L -	Move the display LEFT one half of the screen width.
COL R -	Move the display right one half of the screen width.
COL —n -	Move the display left "n" columns.
COL +n -	Move the display right "n" columns.
DCB -	Display the DCB attributes of the edited dataset.
DI -	Display the last EDIT command entered through FSM.
E (F) -	Terminate FSM, and optionally free the buffer area.
EM (c) -	Enter edit mode and optionally change the trailing blank character for variable length records.
FILL (n) -	Set the fill column to n (default of last).
HB (n) -	Move the display backward n half-pages of data.
HEX (n) -	Display the data in the line display in hexadecimal mode.
HF (n) -	Move the display forward n half-pages of data.
MD (n) -	Move the display to line n, or before if non-existant.
MDR (n) -	Move the display to line n, and remember current line.
NOFILL -	Set the fill column to 0.
PB (n) -	Move the display backward n pages of data.
PF (n) -	Move the display forward n pages of data.
RESET -	Reset all pending line commands and messages.
SM (c) -	Enter scan mode and optionally change the trailing blank character for variable length records.
TABS ON -	Allow tab processing to be done by FSM.
TABS OFF -	Disallow tab processing for FSM.

TECHNICAL SUPPORT

Programming support, enhancements and technical assistance are included in the lease price. Under a Fully Paid License, support is included at no charge for the first year and available for a nominal fee thereafter.

TRIAL PERIOD

TONE SOFTWARE CORPORATION offers the customer a 30-day trial period to evaluate FSM. If at any time during this trial period the user decides not to continue the evaluation, the system and documentation may be returned without obligation.

TONE

Tone Software Corporation
[illegible]
[illegible]
[illegible] 714 991 9460
[illegible] 181532

TPAN INTERACTIVE PANVALET COMMAND PROCESSOR

OVERVIEW

TPAN, Interactive PANVALET Command Processor, provides the user with the ability to fully utilize the facilities of the PANVALET system interactively in a TONE or TSO environment. TPAN enables the user to access PANVALET libraries as well as store, retrieve, copy or delete datasets using a free form command structure. System and programmer efficiency are improved through the use of the following facilities.

A reduced requirement for disk space by storing modules not currently needed on the PANVALET master file in a compressed format.

Simplified management of PANVALET system by providing the data manager terminal access to the directory.

Simplified programmer use by providing free form input and extensive prompting.

Encourages the use of common routines which can be stored in a PANVALET library and retreived by any authorized programmer.

COMMAND FUNCTIONS

Commands are contained in TPAN which allow the user to:

- Add datasets, supersets, and/or subsets to a PANVALET library
- Update datasets, supersets, and/or subsets in a PANVALET library
- Delete subsets from a PANVALET library
- Copy a dataset, superset or subset in a PANVALET library
- Retrieve a dataset, superset and/or subset from a PANVALET library for subsequent editing with the powerful TONE/TSO editor
- Rename a dataset, superset and/or subset in a PANVALET library
- Select parts of datasets from a PANVALET library into a TONE/TSO dataset
- Change the attributes of a dataset, superset or subset in the PANVALET library including
  - Comment
  - Format
  - Status
  - User code
- List dataset, superset or subset information from a PANVALET library directory
- Switch from one PANVALET library to another
- Obtain HELP command support

TPAN commands or subcommands may be issued in any sequence.

PROGRAMMING SYSTEMS

TPAN is supported under the current releases of TONE 3, TONE 4 MVS and MVS TSO.

TECHNICAL SUPPORT

Programming support, enhancements, and technical assistance are provided for the term of the initial contract. Thereafter, support will be available from Tone Software Corporation at a nominal charge.

INSTALLATION

While it is recommended that the customer install TPAN, installation is available from Tone Software Corporation on a time and expenses basis.

TRIAL PERIOD

Tone Software Corporation offers the customer 30 days to evaluate TPAN software and associated documentation free of charge and without further obligation.

TONE

Tone Software Corporation
[illegible]
[illegible] 92801
[illegible]
[illegible]

TRIAL AGREEMENT

AGREEMENT, made this __________ day of ________________, 19_____ by and between TONE SOFTWARE CORPORATION (TSC) and ____________ ____ (Customer).

TSC and Customer agree to install ____________________________ software at Customer's __________________________ facility for a 30-day free trial period beginning _________________________ and ending ______________________________________; and

Customer understands that the above-referenced software is a trade secret of TSC and agrees that it will protect the trade secret software and that it will not copy said trade secret software or any portion thereof to make said trade secret software available to third parties.

Should Customer be dissatisfied with said software, Customer agrees to extract said trade secret software from any other software into which it may have been merged and return all software and transmittal media to TSC and destroy any other form of said software at the completion of said trial period. Upon such election, Customer agrees to certify in writing that all aforementioned software has been returned or destroyed.

Should Customer for any reason fail to return or destroy said software at the end of said 30-day trial period, Customer agrees to commence paying the then current charge applicable for the rental of said software on a month-to-month basis, until such time as a written Lease or License Agreement is entered into between Customer and TSC.

____
Company

____
Signature

____
Name typed or printed

____
Title

TONE
Software Corp.

1735 South Brookhurst • Anaheim, CA 92804 • (714) 991-9460 • Telex 181592

TONE
Software Corp.

TONE PRODUCTS AGREEMENT

AGREEMENT, made this ______ day of ________________________________, 19____, between TONE SOFTWARE CORPORATION (TSC) and ________________________________ (Customer), is made with reference to the following:

WHEREAS TSC is the owner of certain proprietary computer programs and related information; and,

WHEREAS Customer desires to make use of specified TSC products under the following conditions and covenants,

THEREFORE, it is agreed as follows:

LICENSE TO USE

1. TSC grants to Customer for the period of this contract and any extension thereof, a non-transferrable, non-exclusive right to use the " licensed software" at its ________________ "facility", and no other, at the charges specified below:

Item	Term	Monthly Charge	Fully-paid License
TOTAL:			

"Licensed software" will include the programs and all related documentation.

"Facility" shall be defined as a particular data center or computer operations center in a single, specific location.

TRIAL PERIOD

2. Upon receipt of an executed copy of this Agreement, TSC agrees to furnish the licensed software without cost to Customer for a trial period commencing ____________________ and ending ______________________________. Customer understands that it will incur no liability under this contract if during said trial period Customer manifests its intention not to complete this contract by returning said licensed software and related documentation to TSC or certifies in writing that the licensed software has been destroyed. Provided, Customer shall be entitled to one trial period, whether by this agreement, or by a separate trial agreement.

TERM

3. If Customer manifests its intention to complete this contract by failing to return said licensed software, it shall commence paying the charges as shown in Paragraph 1 above on the first day following the completion of the trial period. Customer shall continue to pay the monthly charges for_________ months (initial term), and may thereafter continue to lease said licensed software on a month-to-month basis at the then current monthly charge. Monthly charges will be invoiced in advance and will be payable within thirty (30) days after the date of invoice. Charges for a partial month's use will be prorated based on a thirty (30) day month.

CONDITIONS OF CANCELLATION

4. During the initial term, TSC cannot cancel this agreement or increase the monthly charge as long as Customer is not in default of this agreement. Any monthly installment which is not paid within 30 days of the date of invoice shall be subject to a 10% late charge. In addition, TSC reserves the right to cancel this agreement if any invoice becomes sixty (60) days delinquent. Any reinstatement of this contract following such default shall be at the option of TSC, and in all cases the contract will only be reinstated according to the then current charges, terms and conditions. Customer may cancel this agreement at any time following the initial term upon thirty (30) days written notice to TSC.

WARRANTY

5. TSC warrants that said licensed software will meet its published design objectives and specifications and agrees to promptly correct without charge any faults or inaccuracies which arise solely in conjunction with the utilization of said software. Customer agrees to provide TSC representative with machine and/or personnel resources as may be required to solve problems. TSC further agrees to service said software and provide consultation to Customer with regard to problems which arise in connection with Customer's modification, or provide technical support for enhancements requested by Customer when Customer agrees to pay TSC's normal hourly rate from time to time in effect for such service or support. TSC, however, disclaims any warranties express or implied concerning the effectiveness of any modifications to said licensed software and assumes no liability for loss, personal injury, death or property damage arising out of the use or application of said licensed software. THIS WARRANTY IS IN LIEU OF ANY OTHER WARRANTY EITHER EXPRESSED OR IMPLIED.

PROGRAM SUPPORT

6. TSC will provide system and program technical support at no cost to Customers leasing the licensed software and for the first 12 months to all Customers who acquire a Fully Paid License (FPL). TSC shall continue to provide such support following the first 12 months for Customers who acquire an FPL for a dollar sum annually equal to 15% of the then current FPL rate for the licensed software unless Customer notifies TSC in writing not to provide such support not less than 30 days prior to the expiration of the initial 12-month period.

CUSTOMER RESTRICTIONS

7. Customer shall not copy, in whole or in part, any licensed software materials which are provided by TSC under this Agreement except to : (a) Modify the licensed software as provided below, or (b) Provide for operation and backup. The Customer agrees to maintain appropriate records of the number and location of all copies made. The original, and any copies of the licensed software, in whole or in part, which are made, shall be the property of TSC.

MODIFICATION OF LICENSED SOFTWARE

8. Customer may modify said licensed software for its own use and merge it into other program material to form an updated work, provided that, upon discontinuance of the license, the software supplied by TSC will be completely removed from the updated work and dealt with under this Agreement as if permission to modify had never been granted. The updated work, or any portion thereof, shall be used only at the designated site. Customer agrees to forward to TSC the modified or updated software and such modified or updated software may be used by TSC for any purpose without charge.

INSPECTION OF CUSTOMER SITE

9. Title to all licensed software and any reproduction thereof supplied to Customer by TSC remains with TSC and Customer agrees to return all such software to TSC within thirty (30) days after termination of this Agreement. Customer further agrees that it will certify that such software has been returned and that TSC shall have the right to inspect Customer's facilities within six months following the termination of this Agreement in order to verify that all licensed software has been returned. TSC agrees that any such inspection will be conducted during normal working hours at a time which will not unreasonably interfere with the tasks to be performed by Customer's equipment and/or personnel.

INDEMNITY - PATENT AND COPYRIGHT

10. TSC will defend, at its expense, any action brought against the Customer to the extent it is based on a claim that licensed software infringes upon a United States Patent or Copyright, and pay any costs and damages finally awarded against Customer provided, however, that TSC's liability shall be limited as follows:

(a) TSC shall in no event be liable for damages in excess of $20,000.00

(b) TSC shall have the right to procure for Customer a license to continue using said licensed software, or to replace or modify said software to make said licensed software non-infringing. TSC shall have no liability for any claim based upon the continued use of unmodified infringing software after notice in writing by TSC to cease using the infringing software

(c) TSC shall have no liability for any infringement which results from the use of the licensed software in combination with any other proprietary product

(d) Customer shall promptly notify TSC in writing of the claims and TSC shall have sole control of the defense and all related settlement.

RISK OF LOSS

11. If any licensed software is lost or damaged during shipment, TSC will replace the licensed software and program storage media at no additional charge to the Customer. If any licensed software is lost or damaged while in the possession of the Customer, TSC will replace the licensed software at a charge for program storage media, unless it is provided by the Customer.

PROTECTION OF TRADE SECRET

12. Customer agrees that said licensed software and any enhancements are a trade secret of TSC. Customer agrees that it will not, at any time either during or after this agreement, make available to third parties said licensed software or any information which may be construed to be an improvement on, or a derivative of, the licensed software or related information. Customer further agrees that it shall take reasonable precautions to assure that this Agreement is binding upon all its employees present or future, and enforceable against them. Customer's modifications of any part of said licensed software shall be preserved in confidence to the same extent as the original licensed software supplied by TSC, and the modified software shall be governed by the terms of this agreement.

ATTORNEY'S FEES

13. In the event any party to this Agreement brings legal action or commences litigation or employs legal counsel to enforce any provision herein, the prevailing party shall, in addition to any other remedy afforded it, be entitled to reasonable attorney's fees as determined by the court.

GOVERNING LAWS

14. The validity, construction and performance of this agreement shall be governed by the laws of the State of California, excluding that body of law applicable to choice of laws, and if any provision of this agreement or application of provision of the agreement shall be held by a court of competent jurisdiction to be contrary to law, the remaining provisions of this agreement shall remain in full force and effect.

IN WITNESS THEREOF, the respective parties have hereunto affixed their signature the day and year first above written.

CUSTOMER:__________	TONE SOFTWARE CORPORATION
By:__________	By:__________
Name typed or printed	Name typed or printed
Title	Title
Date Executed	Date Executed

EFFECTIVE
SEPTEMBER 26, 1980

TONE PRODUCT PRICE LIST

PRODUCT	MONTHLY LEASE	FULLY PAID LICENSE
TONE 3	$825.00 per mo.	$26,500.00
TONE 4	$825.00 per mo.	$29,900.00
FSM	$165.00 per mo.	$ 4,000.00
FSO	$200.00 per mo.	$ 7,500.00
328X	$150.00 per mo.	$ 4,500.00
ISP	Not available	$11,500.00 or $1,500.00 for 12 mo.
PHASOR	$225.00 per mo.	$ 6,500.00
TLIB	$ 75.00 per mo.	$ 2,500.00
TPAN	$ 75.00 per mo.	$ 2,500.00
TYPE 3 LIBRARY	$200.00 per mo.	Not available

- Full maintenance is included in all leases.
- A minimum term of one year is required for all leases. Extended term discounts are available.
- One year of maintenance included on all Fully Paid Licenses. Maintenance is optional thereafter at a cost of 15%/year of the then current Fully Paid License price.
- Discounts are available for installation at multiple sites.

TONE
Software Corp.

1735 South Brookhurst • Anaheim, CA 92804 • (714) 991-9460 • Telex 181592

WYLBUR

On-Line Business Systems, Inc.
115 Sansome Street
San Francisco, California 94104
(415) 391-9555

WYLBUR[T.M.*]

. . . A BETTER WAY

- **Program Development On-line**
- **Interactive Text Editing**
- **Remote Job Entry**
- **Cardless Production**

WHAT IS WYLBUR?

WYLBUR is a conversational system that allows users located at interactive terminals to:

- Create and interactively manipulate text using one of the most powerful text editors available. Text is stored in and retrieved from standard OS data sets.
- Submit jobs directly to the operating system for execution in the batch environment.
- Control the execution of jobs submitted through WYLBUR.
- Retrieve at the terminal all or specific parts of spooled job output including the job log, JCL messages, and output associated with specific SYSOUT DD statements.
- Use the WYLBUR EXEC language features to create interactive routines to perform data entry, conversions, calculations or any frequently used process.
- Query the system directly for information contained in the catalog, disk volume tables of contents (VTOC's) and partitioned data set directories.
- Develop word processing applications for documentation, proposals, mailing lists, you name it.

* WYLBUR is a trademark of The Board of Trustees of The Leland Stanford Junior University

WHY DO YOU NEED WYLBUR?

- If you would like to decrease system development time by dramatically increasing staff productivity
  — YOU NEED WYLBUR
- If you would like a "cardless" production environment to simplify job submission, save time and minimize control errors
  — YOU NEED WYLBUR
- If you would like to overhaul your procedure library or clean up that old 1401 application
  — YOU NEED WYLBUR
- If you would like to significantly reduce average test turnaround time
  — YOU NEED WYLBUR
- If you would like to have the power of your computer anywhere there is a telephone
  — YOU NEED WYLBUR
- If you would like to streamline and automate your administrative document processing
  — YOU NEED WYLBUR
- If you have a conversion coming up that requires many source programs or JCL sets to be modified
  — YOU NEED WYLBUR
- If you would like to develop computer-aided-instruction courses for users or your own staff
  — YOU NEED WYLBUR
- If you don't want to spend a lot of money to obtain and run an interactive text manipulation/RJE system
  — YOU NEED WYLBUR

EVERY TEXT EDITOR IS "ONE OF THE MOST POWERFUL" — WHAT DOES WYLBUR HAVE TO BACK UP ITS CLAIM?

WYLBUR text manipulation commands are used to create or modify source programs, documentation, correspondence, test data, job control statements (JCL) or other text. It's easy for technical or non-technical personnel to become effective WYLBUR users with a few hours of training because it uses English-like, free format commands. One simply backspaces over typing errors and retypes. WYLBUR will reject most input command errors with a short but informative message or will prompt for omissions. A command, like any piece of text, can be edited or modified, thus eliminating the need to completely retype commands—even those which the system has rejected as erroneous.

Editing commands operate on a working or 'active' file for each concurrent user. Text from other data sets may be copied, in whole or in part, into the active file. Working files may contain up to 10,000 lines, depending on the density of the text. All lines of text may be referenced either by line number or by text content. Line numbers may contain up to 8 digits (not included in line length) for easy insertion of large groups of text. The working file may also be re-numbered at any time during a WYLBUR session.

Text manipulation features include:

- Upper and lower case
- Full support of "tabs" for easy text collection and rapid list back
- Compression of stored text (user option)
- Storage of text in standard OS data sets (sequential and partitioned)
- Manipulation of single lines, ranges of lines, entire text
- Ability to reference column positions within lines
- Full text recovery after system or line failures

- Prompting after erroneous commands for correction/reentry
- Grouping of commands so that they may be stored and called as a unit for easy execution of common routines
- Use of abbreviated commands (optional for experienced users)
- Variable line length up to a maximum of 133 characters
- Global search or replacement of character strings
- Copy and move functions
- Keyword protection against file alteration and unauthorized use of computer resources
- Off-line printing of selected portions or of entire text files
- Ability to access ordinary OS data sets whether or not they were created via WYLBUR.

HOW DOES THE USER SUBMIT AND CONTROL JOBS?

The user creates a set of JCL (by copying a procedure library member and then modifying the copy, for example) to compile and/or execute one or more programs. The user may then use the following WYLBUR commands to submit and control his job (Users cannot affect jobs submitted by others):

RUN	— Submit job
HOLD	— Place job in system hold status
LOCATE	— Show job status: Awaiting Execution, Executing, Awaiting Output, Awaiting Fetch, Not Found
RELEASE	— Release job from hold status
CANCEL	— Cancel an executing job
PURGE	— Cancel job output
ROUTE	— Assign spooled output to remote or central printers
FETCH	— Retrieve spooled output at the terminal

HOW DOES IT WORK?

WYLBUR executes as a conventional job under OS or OS/VS on IBM 370 systems. WYLBUR may run 24 hours per day or be terminated when its services are not required. An idle system consumes almost no resources. It must run at a high dispatching priority in order to ensure good response time (normally one second or less). Because WYLBUR was designed for maximum efficiency, it can run at high priority, provide excellent response time to users and have very minimal impact on batch throughput. For example, 30 to 50 concurrent users can enjoy excellent response time in 384K of virtual storage on a 370/158 without hampering batch work. On-line applications (CICS, for example) can run concurrently with WYLBUR.

HOW MUCH DOES IT COST TO OPERATE?

After a data processing manager discovers what can be accomplished with this powerful system, he or she is ready for the most pleasant surprise of all — the cost to obtain and operate WYLBUR. While processing costs vary with the number of concurrent users and computing resources available, we are confident you will agree that WYLBUR is inexpensive to operate. For example, CPU utilization averages 16-20 CPU seconds per connect hour on the OBS 370/158, Model 1. Further details on resource utilization are available upon request.

We urge you to compare WYLBUR to TSO and other systems of similar nature. Compare features. Compare response time. Compare ease of installation. Compare maintenance costs. Compare price. Compare operating costs. If you are not firmly convinced about the superiority of WYLBUR, try it under our rental plan. One-hundred percent of your rental payments, up to a maximum of seven months, can be applied toward a perpetual license.

WHAT ARE THE HARDWARE REQUIREMENTS?

WYLBUR can be installed on 370 systems (model 135 and larger) or plug compatible CPUs under OS/MVT, VSI, SVS, MVS (HASP or JES). A transmission control unit such as the IBM 270X or 370X is necessary for dial-up or remote terminals. A 3272 Control Unit is required for locally attached 3277 terminals.

Terminals supported include local and remote 3270-compatible devices, 2741-compatible devices and the expanding and inexpensive variety of TTY-compatible terminals in either hard copy or CRT versions.

WHERE DID WYLBUR ORIGINATE?

WYLBUR was developed by computer scientists at Stanford University's Center for Information Processing. Despite the lack of any organized marketing or distribution effort in the past, the number of installations using WYLBUR has grown to over one hundred in the U.S., Canada and Europe. This occurred as students, faculty and visitors, having once used WYLBUR, spread the word about its powerful capabilities. As a result, requests for the program came to Stanford from a large number of organizations. Stanford subsequently licensed other educational institutions on a do-it-yourself basis without providing installation, maintenance or other support.

In late 1976, On-Line Business Systems, Inc., concluded an agreement with Stanford whereby OBS acquired the rights to commercially market WYLBUR. Now, complete support for WYLBUR, including installation, training, documentation, enhancements and maintenance service, is available.

Development of WYLBUR started at Stanford in 1967. OBS has been providing WYLBUR service from its data center since 1974. Consequently, WYLBUR is a well seasoned product, having been enhanced, tuned and polished over the years.

WYLBUR is available under monthly or permanent license arrangements.

WHAT ABOUT INSTALLATION?

Installation is short and simple. No special hooks or Operating System modifications are required. Installation options are available for certain defaults and conventions. User exits are provided at key processing points to allow tailoring by the licensee. Installation will take about one working day. Systems programmer training is provided at installation time.

CAN YOU TELL ME MORE ABOUT OBS?

On-Line Business Systems, Inc., is a privately held corporation formed in San Francisco in 1969 to offer on-line transaction oriented data services. It has developed a multiplicity of real-time applications ranging from reservations to order entry systems. These applications operate on an IBM 370/158 in the OBS data center over a nation-wide teleprocessing network.

In 1974, OBS management set out to find an inexpensive way to bring the power of the computer closer to the development staff and to lower applications development costs. Stanford University's WYLBUR was one of several terminal-oriented systems studied and was selected by OBS for installation. Since then, WYLBUR has had an amazing impact on the efficiency of our data center and on the productivity of OBS' staff and of the personnel of our clients using the system. We are pleased and proud to have the opportunity to make WYLBUR available to many new users.

HOW ABOUT MORE INFORMATION ON WYLBUR?

If you would like more information about WYLBUR, call us at (415) 391-9555 or write to:

On-Line Business Systems, Inc.
115 Sansome Street
San Francisco, California 94104

EVALUATING WYLBUR AND ALTERNATIVES

EDITING POWER

The methodology provided by a product that allows a user to rapidly locate and easily change specific text items is the foundation for productivity improvement. The Window Mode Full Screen Editor works hand in hand with WYLBUR's line range concept, to make editing a breeze. The CONTEXT, DENSE, CARD and LONG display options provide users with multi-approach editing tools for variable needs. The FSE window concept is also the perfect solution for reviewing 132 column job output on an 80 column terminal. With a little experience, your programmers can minimize editing time thru the WYLBUR DEFINE command and PF key support. After working with this powerful tool, you will be convinced there is no better implementation of full screen edit in any program development/text editing system.

USABILITY

WYLBUR was installed in over 100 installations before it was a supported product. The major reason for this incredible popularity stems from WYLBUR's ease of use. English language syntax coupled with a forgiving command structure make it simple for users to learn. Although WYLBUR offers a large number of commands and options, it is the kind of system which allows users to be effective with knowledge of a small subset of the possible variations. This is one of the reasons WYLBUR is fun to use--you can learn something new every day and never get into trouble by experimenting.

RESPONSE TIME

Good response time is an important factor for productivity improvement. Poor response time results in user frustration and impatience. Obviously, the better the response time the more a user can accomplish in a given amount of wall clock time. Ask our competitors what good response time is. Don't be surprised if the answer is 10-15 seconds. Ask a WYLBUR user what response time is like and you will find that our users are accustomed to 1-3 seconds. This kind of response is possible even on heavily loaded systems because WYLBUR's architecture was designed for maximum efficiency. This means excellent response time is possible without significant impact on other tasks competing for resources in the same CPU.

Editing power, usability and response time are key factors in maximizing productivity. No system does a better job than OBS WYLBUR in these three critical areas. Of course, there are other things one must consider when selecting a product such as WYLBUR. Here are some other areas which were important in past evaluations.

RESOURCE CONSUMPTION

Although hardware costs are decreasing, system resource requirements are still important. WYLBUR has always had a reputation for low overhead and although OBS has added major features to WYLBUR, we haven't neglected performance. Performance is especially critical to good response time and also important when the host environment is in the 370/135 to 148 range. You will not find any system with features comparable to WYLBUR requiring less CPU resources. Here are the facts taken from our own 370/158 Mod 1:

Peak Daily Concurrent User Load	50
Total CPU Time per Day including all overhead	2 Hours 18.48 sec/hr

ACCESS CONTROL

Please refer to the description of the OBS WYLBUR Access Manager in the Systems and Operations Guide and in the Reference manual. This extended security feature provides an excellent safeguard against unauthorized access to datasets and jobs.

APPLICATION IN THE DATA CENTER

Often neglected by our competitors (because most software vendors have no experience in running a production-oriented data center) is the Operations department. OBS offers a working adaptation of WYLBUR capabilites to solve many of the day-to-day problems experienced in the data center. The enclosed Introduction to OBS WYLBUR offers a concise explanation and schematic configuration of the OBS Operations Support System (OSS). OSS is attractive to many data center managers because it is a straight-forward, easy to implement production support system. If OSS is implemented by your organization, the data center can provide additional justification for acquisition of WYLBUR.

USER GROUP

An important development for OBS WYLBUR has been the introduction of a formal OBS WYLBUR User Group. Known as COLLECT, the group was established during the March 1979 SHARE Convention and is open to all OBS WYLBUR installations. The User Group functions independently of OBS with an elected steering committee and an OBS advisor (in a non-voting capacity). The group meets twice a year, and through the steering committee submits a formalized list of requested enhancements. In addition to product enhancement, the user interchange has resulted in the development of resources such as documentation, field tested Execute Programs and a user profile available to each installation. The profile identifies similar installation environments for discussion of user developed mods, common problems and new ideas.

Consequently, enhancements to OBS WYLBUR are determined in cooperation with the users. Our users have an established vehicle in which to approach their vendor with enhancement requests, constructive criticism and valuable comments. You will find that OBS is sensitive to the needs of our installations when it comes to development of the WYLBUR product.

PRODUCT SUPPORT

Currently, there are five systems programmers at OBS who are qualified to provide WYLBUR maintenance support. The WYLBUR product knowledge at OBS is not centered in one or two individuals. Our system is well documented, which allows us to bring new people in and make them rapidly productive. If you call with a problem, you can expect fast response from a qualified person who knows the product. This support is available 24 hours per day through our data center. Direct dial-up maintenance support is available if the installation's Telecommunications Control Unit supports dial-up devices. Although rarely needed, this dial-up access can be very helpful in diagnosing and correcting problems.

At OBS, we realize that the future of our product is dependent upon satisfied customers. Our current customers have received major enhancements to WYLBUR (Window Mode FSE, Access Manager, DEFINE command, 232 line length, etc.) at no additional cost to them while our competition generally offers new features as extra cost items. If you become one of our customers, you can look forward to a product which gets better with age.

If there are other areas important to your evaluation that are not discussed here, please let us know. We urge you to take a hard look at the areas that are fundamental to successful productivity improvement. Make sure that you don't base your decision on a list of superficial features which you may not need or which have been provided at the expense of solid fundamentals.

In summary, if you desire:

- A quality product that will please its users--
- Product support from a substantial company--
- Thoroughly designed and carefully implemented product improvements--
- A system which minimizes its use of resources--
- A system which is priced right--

Then you can't go wrong with OBS WYLBUR.

TRACK RECORD ON ENHANCEMENTS

Our track record speaks for itself. OBS began marketing WYLBUR in January, 1977. At that time, we had a clean implementation of WYLBUR for SVS/HASP environments. During 1977 we completed the following enhancements:

- Implementation of WYLBUR under MVS/JES2 without SCP mods.
- Implementation of WYLBUR under VSI/JES1 without SCP mods.
- PAGER, a modularized document formatting facility.
- Implementation of user exits.
- Implementation of SYSDEFN, an installation option macro.
- User defined dataset naming conventions.
- Profile (or LOGON) Exec files.
- Logical Tab support for terminals without hardware tabs.
- Basic Mode 3270 support via EXCP for local terminals.
- System Console support.
- Reductions in resource consumption.
- Operations Support System (an optional feature of WYLBUR).
- Graphics terminal support.

During 1978, we completed the following enhancements:

- Implementation of WYLBUR under OS/MVT with or without HASP and without SCP Mods.
- Basic Mode 3270 support via EXCP for remote terminals.
- 3284/3286 local or remote printer support.
- Communicating Mag Card typewriter support.
- Implementation of dynamic network control capabilites and sophisticated network error recovery capabilities.
- NOTIFY capability.
- ADR LIBRARIAN interface.
- Window Mode Full Screen Edit.
- User defined Program Function Keys for 3270's.
- System installation by the user.
- SCRIPT support for total document formatting capabilities.
- User defined commands for all terminal types.
- DASD Access Manager.
- Controlled volume selection on active file recovery.
- Simplified and streamlined subsystem interface to support userwritten subsystems under WYLBUR's TP Monitor.
- Further reductions in resource consumption.
- ON ERROR Command (A "Break Exit" for EXEC files).
- Active file option for all WYLBUR SHOW commands.

During 1979, we completed the following enhancements:

- VTAM Support (Local, Remote Bisynch & SDLC Protocols)
- Alternate JES2 Interface providing increased functions
- 3728 terminal support (Models 1-5)
- 232 line length
- MILTEN trace facility
- Extended PF key support through 24
- NJE Support
- PANVALET Interface
- New User Guide including Full Screen Edit Tutorial

During 1980, we plan to continue an aggressive schedule of significant enhancements. Some of the proposed enhancements are listed below:

- JES3 interface
- Dynamic system generation
- Implementation of DSN macro in Access Manager
- ACF2 Interface
- User Security Exit at Dataset Level
- Exit for user written commands
- User Defined JOB Statements
- RENAME Command
- RESAVE Command
- Dynamic volume allocation
- Multiple commands per line

- VSAM catalog support
- Implement Window Mode Commands in Exec Files

Our motivation to produce these enhancements is tempered with our knowledge of the work required to produce high quality systems. In this manner, our customers are assured that new features will not be produced at the cost of a stable and maintainable product.

3rd Edition
June 1980

INTRODUCTION TO OBS WYLBUR

Additional copies of this manual may be acquired by contacting:

ON-LINE BUSINESS SYSTEMS, INC.
115 Sansome Street
San Francisco, California 94104
(415) 391-9555

TABLE OF CONTENTS

INTRODUCTION

This document provides a thorough overview of WYLBUR, its functions, implementation requirements, performance, documentation and the support for WYLBUR provided by OBS. It is intended for use by customers licensed to OBS WYLBUR, and prospective customers who are evaluating OBS WYLBUR.

History.

WYLBUR was developed by computer scientists at the Stanford University Center for Information Processing. Despite the lack of any organized marketing or distribution effort in the past, the number of installations using WYLBUR has grown to over one hundred in the U.S., Canada and Europe. This occurred as students, faculty and visitors, having once used WYLBUR, spread the word about its powerful capabilities. As a result, requests for the program came to Stanford from a large number of organizations. Stanford subsequently licensed other educational institutions on a do-it-yourself basis without providing installation, maintenance or other support.

In late 1976, On-Line Business Systems, Inc. concluded an agreement with Stanford whereby OBS acquired the rights to commercially market WYLBUR. Now, complete support for WYLBUR, including installation, training, documentation, enhancements and maintenance service is available.

What is WYLBUR?

First and foremost a program development system, WYLBUR also provides document formatting capabilities and EDP production support capabilities to OS and OS/VS installations. By providing text editing, RJE/RJO, document processing, dataset utility functions and even an interactive procedure language, WYLBUR provides users with "open-shop" access to the computer without causing the management problems normally associated with an open-shop. It offers "big shop" program development capabilities to smaller 370 installations and also offers a realistic vehicle to supplement or replace TSO in medium to large installations experiencing problems supporting the resource requirements of TSO.

Functions

Text Editing - Using English language commands, WYLBUR offers powerful editing facilities. In addition to common line oriented commands (DELETE, INSERT, REPLACE, MOVE, COPY) WYLBUR offers sophisticated global commands for searching and/or changing specific character strings. Searches may be specified by line number, text content, column positions, absence of a string, string occurrence or any combination of the above. Editing commands operate on an ACTIVE or working file for each concurrent user. Maximum ACTIVE file size is variable but 10,000 lines of text is a practical limit. Maximum text line length is currently 232 characters. ACTIVE files are automatically recovered in the event of any system or communication line failure.

WYLBUR user datasets are standard OS datasets, sequential or partitioned. Space compression is available at the user's option. Datasets are dynamically allocated and space accounting functions prevent excess disk space allocations by users. A read-only interface to LIBRARIAN is provided to LIBRARIAN licensees. A similar interface to PANVALET is also provided to PANVALET licensees.

RJE/RJO - Users may submit jobs from a terminal to run in batch mode. Once submitted, the job may be controlled by its owner (i.e., Held, Released, Cancelled, Routed, Purged, Printed). When the job completes, the spooled output may be reviewed by the user at the terminal. All text editing commands may be used to search for and display desired job output lines. After reviewing spooled output, the user may decide to print or purge the output listing.

Dataset Utilities - VTOC inquiries, Catalog inquiries and PDS directory inquiries are available to users as well as SCRATCH, CATLG, UNCATLG, RECATLG functions for "owned" datasets. Privileged functions are available to allow certain users to bypass ownership requirements.

Document Processing - Editing commands coupled with a special interactive formatting program allow: text justification (ragged right or left/right justified), pagination, centering, indentation, page numbers, headings, footings and table of contents or index generation. Form letter generation with stop code capability is also available.

Procedure Language - A WYLBUR Execute Program is simply a dataset containing one or more WYLBUR commands which may be called as a unit and executed in a controlled sequence. This effectively provides a powerful, interactive programming language. Functions available include: Terminal I/O, String processing commands, External File I/O, Branching, Branch and Link, XCTL capability to pass control to subsequent EXECS, and a pre-processor that will scan commands and provide text substitution based on a symbolic parameter concept. Execute programs provide an excellent vehicle for JCL or Source conversions, Computer Assisted Instruction, JCL creation, User profiles or any low volume input interactive application.

Operations Support System - Available as an optional feature of WYLBUR. OSS offers automated production job scheduling and submission, facilities for creation, inquiry, update and reporting on an on-line production job documentation library; automated job restart facilities, and audit trails to facilitate research of production problems. OSS eliminates card handling errors, provides current documentation to the operations staff on a daily basis, simplifies training of new personnel, provides a production job acceptance vehicle and helps eliminate data center dependence on "application experts."

What Will WYLBUR Do For Me?

Improve Productivity!

In a purely batch mode of operation, much of the effective, productive time of people is wasted waiting for results and wading through large volumes of printout.

Programming time, and particularly program maintenance time, can be dramatically improved with on-line editing and remote job management capability. Many tasks which previously took weeks or months can be completed in a few days with the right program development tools.

The workload on the Operations department is reduced significantly when WYLBUR is installed. Although the number of jobs submitted generally increases, Operations will actually need to handle fewer jobs since no input decks are needed and much output is never printed because the job output was reviewed at the terminal.

Less departmental interaction between Operations and Programming or other groups results in more productive use of each group's time.

Users can work on one task to its completion, or at least to a logical breakpoint, instead of being restrained by poor turnaround. This means fewer mistakes because of concentration on one set of problems. It also results in faster project completion and an earlier return on investment.

Improve Working Conditions

We all know how tough it is to find and keep good technical talent. As with any job, the better tools you provide your employees, the better job they will do and the more they will enjoy doing it. If WYLBUR is nothing else, it is incredibly popular with its users. This is usually because of fast response time and WYLBUR's ease of use. With WYLBUR, your installation will be a better place for your staff to work.

Simplify Operations

By eliminating card decks, reducing printed output and providing a vehicle for programmers to determine job status, you eliminate several functions that required people time (and thus management time) in the past.

Minimize Resource Requirements

Unlike most competitive systems, WYLBUR was designed for efficiency. Through advanced system architecture and efficient code, WYLBUR will cost less in terms of CPU time than any competitive system supporting a similar number of concurrent users.

HARDWARE-SOFTWARE REQUIREMENTS

WYLBUR is written in Assembler language code. Its input/output routines use EXCP-level programming. The system (including its built-in TP monitor) requires one region/partition to run. Generally, all the equipment used by WYLBUR is already required by the operating system. Specific hardware/software requirements follow:

WYLBUR HARDWARE/SOFTWARE REQUIREMENTS

REQUIREMENT	TYPE
CPU	IBM S/370 Model 135 or larger, or equivalent compatible device
Main Storage (minimum)	512K bytes real storage on the CPU (WYLBUR requires a minimum 256K virtual region)
Operating Systems Supported	OS/MFT, MVT, VS1, VS2 (SVS, MVS)
Spooling Systems Supported	HASP, JES1, JES2, JES3
Terminals	2740, 2741, TTY, 3270 (local and remote), 3278 models 1 through 5, or equivalent compatible devices
Direct Access Storage Devices (for WYLBUR load library and user dataset storage)	All IBM-compatible direct-access storage devices (2311, 2305, 2301, 2314, 3330, 3330-11, 3350, 3340, etc.)
Tape Drive (for loading system)	2400/3400 series compatible tape drives (9-track)
Printer (for high-speed listings and job output listings)	1403 (with TN-train if upper/lowercase printing required), 3211, or equivalent devices

FUNCTIONS AND FACILITIES

What Happens in a WYLBUR Session?

A session is a series of prompts, commands, and responses. WYLBUR asks for a command--you issue a command. WYLBUR prompts with a line number for text input--you type in the text you want. If more information is needed to process a command, WYLBUR often asks you for it rather than simply rejecting your command.

WYLBUR also has a RETRY mode which allows you to modify commands that fail. WYLBUR gives you meaningful error messages when commands are incorrect, and informational messages when commands succeed.

WYLBUR has a powerful set of commands to handle its functions. Powerful does not mean complicated. The commands are well thought out English words which are easy to learn, and most commands can be abbreviated to their first three letters. You need know only a few commands to be productive immediately. This means you can use WYLBUR effectively with very little formal training.

What Kinds of Things Can I Do With WYLBUR?

Creating Updating, Editing Datasets	A WYLBUR session begins by signing on to the system. Each person who signs on to WYLBUR is assigned a personal work area called an ACTIVE file. Lines of text are automatically numbered as they are entered into the ACTIVE file. You may make corrections or changes to your data at any time while it is in your ACTIVE file. When complete, you may save this data permanently on disk in a sequential dataset or as a PDS member. Once saved, it can be retrieved for updating, printing, execution, etc. When you save and retrieve external files, you refer to them by the names you have assigned them. The installation may utilize the Access Manager to control the allowable type (Read, Execute, Update) of access to volumes by specific users or classes of users (see the section on security for more information).
Utility Functions	You may want to examine and update the system catalog, look at PDS directories, and see actual dataset statistics such as tracks and directory blocks used, record format, etc., by using WYLBUR commands

Job Management (RJE/RJO)	For remote job management, you use your active file as a work area for creating jobs (JCL) and then holding the output of jobs 'when they complete. You may monitor jobs as they run in the central computer, and also have WYLBUR notify you when your jobs complete.
Word Processing	To produce a formatted document (proposals, contracts, letters or manuals, for example) you use your ACTIVE file as a work area while you tell WYLBUR how to arrange the text. Documents can go straight from your ACTIVE file to the high-speed printer when you are ready.
High-level Programming	You may also use WYLBUR's Interactive Programming Language to write simple programs composed of special WYLBUR commands, and have these programs converse with you at the terminal, perhaps requesting information from you, displaying reports, updating files, submitting jobs automatically, etc.

Will I Like Using WYLBUR?

WYLBUR has been designed for you -- to communicate with you during the terminal sessions, to respond to your requests and to ask you questions when more information is needed WYLBUR commands have short forms (abbreviations) that make them easy to use; an especially nice feature for the non-typist. While signed-on you have a one-to-one relationship with the central computer. It appears as if you are the only user on the system. Everything you ask WYLBUR to do is done independently from anything else that may be going on within the system You may submit jobs, send files to the high-speed printer and begin work on something else -- with each task executing concurrently Response times are exceptionally fast.

WYLBUR also protects your work. Should there be any hardware or software failure, your ACTIVE file is fully protected. You will never spend time re-entering lost text. These features help increase your efficiency and productivity.

Text Entry and Editing

Text entry simply means collecting information for subsequent retrieval Text editing is the manipulation and changes that you perform on that information. Through WYLBUR, you have the computer's speed and power at your fingertips for entering and editing text.

You are connected to WYLBUR through either a hard-copy or CRT-type terminal You enter text by keying the text at the terminal Each line of text you enter is numbered and displayed To make changes to the text, you reference line numbers or character strings, or with

Window Mode Full Screen Edit, you move your cursor to a line and type in new text. You can tell WYLBUR whether or not to display each change after it has been made.

As you become proficient, you can have WYLBUR shorten all its responses for quicker editing.

If you move lines of text around or insert new lines, WYLBUR will keep the original line numbers for the remaining text. Only the moved or inserted sections are renumbered -- by decimal increments.

What Are The Text Editing Commands?

WYLBUR commands are shown in uppercase in the following paragraphs.

To begin, you need to sign-on to start a terminal session on WYLBUR.

• LOGON	tells WYLBUR that you want the sign-on procedure.
• LOGOFF	signals the end of a work session.

To enter text once you are signed on, you tell WYLBUR to:

• COLLECT	lines and automatically number them.
• INSERT	single lines between existing lines.

To move lines of text around, you tell WYLBUR to:

• COPY	lines into a new location without disturbing the originals. Lines to be COPIED may exist in the ACTIVE file or in a separate dataset on disk.
• MOVE	lines from one place in the ACTIVE file to another, deleting the originals.
• REPLACE	existing lines with new text.
• DELETE	existing lines.

To change characters in the text without replacing the entire line, you can either position the 3270 cursor on the lines to be changed and enter the new text, or use WYLBUR commands to:

• CHANGE	one character string to another character string in specific lines or throughout your ACTIVE file.
• EDIT	lines by having WYLBUR list the line and allow you to type only the new characters.
• MODIFY	lines using alteration techniques which allow you to insert, delete, or replace characters.
• UPDATE	specific lines or full screens of text lines via Full Screen Edit.
• UNPROTECT	a full screen of text lines in order to cursor edit a section of your ACTIVE file.

To examine text lines or get a printout of your ACTIVE file, you tell WYLBUR to:

• REVIEW	lines by displaying specific lines or full screens of lines in the Full Screen Edit format.
• LIST	lines by printing or displaying them at your terminal or on the high-speed printer.
• PUNCH	your ACTIVE file to cards.

There are many options available which may be set for a terminal session, and most of these options have a default value assigned by WYLBUR.

Options are defined with the SET command and their values displayed with the SHOW command:

• TERSE/VERBOSE	governs the verbosity of WYLBUR prompts and messages.
• TABS	can be set or displayed, even if your terminal does not have a hardware TAB feature.
• LENGTH	is the maximum number of characters allowed on a line before a warning is issued.
• CASE UPPER/UPLOW	CASE defaults to upper only, but can be set to upper and lower (UPLOW).
• BACK	allows you to use an uppercase backspace to enter a backspace character into your text, and a lowercase backspace to erase the previous character.
• DELTA	is the increment WYLBUR uses in calculating line numbers.
• FASTLIST/ SLOWLIST	tells WYLBUR whether or not to use terminal tabs to speed terminal listings on hardcopy devices.
• RETRY	sets the mode in which WYLBUR will allow you to retry any failing command without having to type it in again.
• SHORT|NOSHORT	determines whether or not line number (immediate) commands are to be verified prior to execution, thereby preventing accidental deletion or replacement of existing line(s) in the ACTIVE file.

Example of Entry and Editing

This example shows a sign-on procedure and a paragraph entered and corrected. In this and all the following examples, the symbol '(ATTN)' is used to indicate the appropriate break feature on a terminal. On 2741 terminals, the key is an 'Attention' key. On TTY terminals, the key is a 'Break' key. On 3270 terminals, the key is a 'PA1' key.

```
OBS WYLBUR REL 5.1 TERMINAL TRMROOM 4 16:53:14 05/01/80
USERID?  ozusr
KEYWORD?
ACCT NO?  2550A22
ACCT IN EFFECT:  2550A2200
COMMAND? set uplow
COMMAND? collect
    1.    ? wylbur is easy to use.
    2.    ? The comands are easy
    3.    ? to remember and I don't need to
    4.    ? know a million commands in order to be
    5.    ? productive right away.  The response time
    6.    ? is very good and I think I can therefore
    7.    ? get more done in less time
    8.    ? (ATTN) ***
COMMAND? change 'wylbur' to 'WYLBUR'
    1.    WYLBUR is easy to use.
COMMAND? collect 1.1
    1.1   ? A powerful set of commands allows me to
    1.2   ? enter and edit text, save and retrieve
    1.3   ? datasets, and run jobs remotely.
    1.4   ? (ATTN) ***
COMMAND? change 'm' to 'mm' in 2
    2.     The commands are easy
COMMAND? list
    1.    ? WYLBUR is easy to use.
    1.1   ? A powerful set of commands allows me to
    1.2   ? enter and edit text, save and retrieve
    1.3   ? datasets, and run jobs remotely.
    2.    ? The commands are easy
    3.    ? to remember and I don't need to
    4.    ? know a million commands in order to be
    5.    ? productive right away.  The response time
    6.    ? is very good and I think I can therefore
    7.    ? get more done in less time
    8.    ? (ATTN) ***
COMMAND?
```

Explanation:

After the sign-on procedure is completed, the SET UPLOW command tells WYLBUR that upper and lower case characters will be entered. Otherwise, WYLBUR automatically converts all alphabetic characters to upper case.

COLLECT is the command used to tell WYLBUR you want to create text.

WYLBUR responds to the COLLECT command with line number prompts. As each line number is prompted, WYLBUR waits for you to type text. When you strike a carriage return WYLBUR prompts with the next line number, and so on. With Window Mode FSE, a full screen of lines may be typed before the ENTER key is pressed.

After text entry is complete, an (ATTN) notifies WYLBUR to stop collecting and return to COMMAND mode. Three asterisks (***) are printed whenever the (ATTN) key is struck in COLLECT mode.

A lowercase string is to be changed to an uppercase string wherever it appears in the ACTIVE file.

Multiple lines are to be inserted after line 1.

One particular character string ('m') is to be changed to another ('mm') in line 2.

LIST tells WYLBUR to print/display lines from your ACTIVE file. You may also list by line numbers or by character strings. (Example: LIST 'ABC' or LIST 2/5).

FULL SCREEN EDITING

In general, Full Screen Editing (FSE) is the ability to modify any character or group of characters displayed on a Video Display Terminal (VDT) simply by positioning the VDT's cursor at the position you wish to modify and typing over the existing text. Special function keys on the VDT further simplify editing because they allow the user to insert a variable number of characters between existing characters, delete existing characters, erase entire lines of text from the point at which the cursor is positioned and rapidly move the cursor to any desired positon on the screen. This capability is especially valuable because much editing is accomplished utilizing the intelligence built into the VDT rather than requiring program logic and user-program interaction. The major benefits accruing from the utilization of FSE are:

> Increased user productivity - Editing is much faster because there is less typing required and less interaction with the editing program. Terminal I/O is only required to transmit full screens of text updates rather than single lines.
>
> Since I/O is reduced, the system can provide even better response time to more users. Overhead is reduced and the system will require fewer CPU resources.

Many systems in today's marketplace offer an FSE feature. There are a number of features provided by WYLBUR's Window Mode FSE that are unique and have accounted for the overwhelming acceptance it has received to-date.

The following summarizes the important features:

WYLBUR's 3270 support consists of Basic mode and Window mode display formats. In Basic mode, commands and responses proceed down the screen to line 23, then resume at the top of the screen. A command input area of 3 blank lines serves as a separator delimiting the most recent response from previous displays. When text lines are displayed in Basic mode (via the LIST command, for example), each text line is displayed on two screen lines as explained below under the window display option LONG.

In general, it is unnecesary to issue a command to switch in or out of Window Mode. The Window is the default display for certain commands while other commands will switch the screen to Basic Mode.

Switching In and Out of Window Mode

> Two commands, REVIEW and UPDATE provide full screen display and display/edit capabilities. These are Window Mode

commands.

The COLLECT and INSERT commands default to Window Mode operation. These commands, however, provide options (NOWINDOW) that allow the user to indicate that he wishes them to operate in Basic Mode.

The DELETE, CHANGE, MOVE, COPY, LINE NUMBER, USE, and FETCH commands work in conjuncton with Window Mode. Commands which replace the ACTIVE file (USE, FETCH) will also cause the new ACTIVE file to be displayed starting with its first line if a Window existed when the command was invoked. The line oriented editing commands will cause the Window to be re-displayed centered around the last ACTIVE file line changed by the command.

The LIST, EDIT and MODIFY commands are Basic Mode commands and will force the screen into Basic Mode when invoked. These commands are not recommended if a 3270 terminal is in use because the REVIEW and UPDATE commands provide much more flexibility and better productivity.

All other commands are Basic Mode commands, but mode switching from Window Mode to Basic Mode is done only when the command entered requires more than a single response line or is otherwise unsuited to operate in Window Mode.

Any switch to Basic Mode causes the screen to be cleared. Command options are provided to allow the user to restrict operating modes to Basic Mode either in general or on a per command basis.

The Window Mode Screen

General Layout

- Line 1 - prompt and command input area.

- Line 2 - response, if any, to last command.

- Line 3 contains a column scale that always corresponds to the text displayed below.

- Line 4-23 constitute the ACTIVE file window and are used to display and update lines of text from the ACTIVE file.

- Line 24 is used, as in Basic Mode, to signal asynchronous events and to contain the range-pending 'MORE ...' indicator.

The Active File Window

- The ACTIVE file window (screen lines 4 through 23) consists of line number fields and text fields. The line number fields are protected. The text fields are protected or unprotected depending on the command that produced the display. The line number fields are omitted for commands specified with the unnumbered option. Screen lines not used are protected.

• Any change to a text field in the window is applied to the ACTIVE file when ENTER is pressed, provided there has been no intervening PA1.

• The line number fields are made as small as possible while still maintaining digit alignment of line numbers and column alignment of text fields. This allows maximum text line length in the window display.

• The portion of the text lines displayed depends on the current column setting of the ACTIVE file window (see the WINDOW and SET WINDOW commands). The scale line (screen line 3) identifies columns shown in the text areas of the ACTIVE file window.

• Padding characters for window text fields are either blank or null characters. The user determines this with the SET WINDOW command.

• Window Formats.

  Three different screen window formats are supported. Each of these formats allow for the display of ACTIVE file lines in a slightly different form. These formats are designated as CARD, LONG and STANDARD Window formats.

  1) STANDARD Window format allows one screen line for each ACTIVE file line. This format allows for a display of a maximum of 79 characters per line with the UNNUMBERED display option in effect and from 70 to 77 characters per line otherwise. Only those columns indicated by the current window setting are displayed.

  2) CARD format requires two screen lines per ACTIVE file line displayed. The WYLBUR line number is displayed on the screen line immediately preceeding the applicable text. This format allows for display of a full 80 characters per ACTIVE file line without splitting the text onto two screen lines.

  3) LONG format also requires two screen lines per ACTIVE file line displayed. This display format allows the display of a full 133 character line with either the NUMBERED or UNNUMBERED options. The text field wraps from the initial line of its display onto the following line.

The SET WINDOW command allows the user to select the Window format preferred.

Considerations For Use

The Window is always representative of the contents of the ACTIVE file. The only point at which the Window display differs from ACTIVE file content is when the user is cursor editing the screen, prior to pressing ENTER. When any command which alters the ACTIVE file is processed, the Window is reformatted either to continue with the specified command range or to display the changed ACTIVE file.

When more than a single Window full of lines must be displayed to complete a command, a 'MORE...' message is placed into line 24.

Pressing PA1 will break the range of a command and cause all modifications of the current ACTIVE file window to be ignored. Current lines are re-displayed to insure that the window remains representative of the contents of the ACTIVE file.

The current command is left on the command prompt line until the command has completed or been interrupted, at which time the command prompt is re-issued with a command field filled per the FILLER option.

Any modification of the current command has the following effects when ENTER is pressed:

- modifications to text in the ACTIVE file window are applied to the ACTIVE file.

- the current command is then aborted.

- the modified command is initiated.

NOTE The WINDOW LEFT, RIGHT and COLUMN commands are unusual in that they do not abort the current command, but merely move the ACTIVE file window so that unseen columns of current lines may be seen or modified without affecting the completion of the current command range

Types of Window Displays

Contextual Displays.

WYLBUR uses the following sequence of logic in order to display a range of lines in CONTEXT:

1) The first line within the command's range is found.

2) This line along with up to three preceeding ACTIVE file lines are displayed on the screen (provided such lines

exist).

3) The remainder of the ACTIVE file window is filled with subsequent, contiguous ACTIVE file lines.

4) All lines in the Window that are also designated by the range are displayed in bright intensity.

5) If the command range requires continuation onto subsequent screens, the next line of the range that is not currently displayed is found and displayed with up to three preceeding ACTIVE file lines.

Dense Displays.

Dense displays consist of only those lines in the command's range.

Command options are provided to allow the user to select display type in general and on a per command basis.

Split Screen Editing

Window Mode FSE provides the ability to display separate areas of the ACTIVE file on a single screen. If for example, the user is reviewing compiler output, he may wish to display the diagnostic messages while also examining the associated source listing. The number of lines to be displayed in each screen portion is totally flexible as long as the sum does not exceed 20.

If for example, diagnostics began on ACTIVE file line number 62 and the source listing began at line number 153, the command:

```
  REVIEW 62(5), 153(15)
```

would display five lines of diagnostics and 15 lines of compiler output on a single screen. To page through the file(s) the user would simply modify the starting line number (62 and 153 in this example) and press ENTER.

The DEFINE command can also be used to establish Program Function Key values The user can therefore DEFINE a PFK to display diagnostic information (or a certain group of lines from a storage dump, for instance) any time the PFK is pressed. This can be very helpful when trying to debug those "unstructured" programs as the user can easily "bounce" back and forth between different areas in the ACTIVE file.

Miscellaneous Considerations.

Modification of lines containing non-representable characters is not allowed using Window Mode This is to prevent inadvertent loss or translation of such characters Window lines containing such characters have PROTECT and SKIP attributes, their line numbers are highlighted, and an explanatory warning message is included on screen line 2. Non-representable characters may be modified using the CHANGE, MODIFY or EDIT commands.

Window Command Synopsis

To use the Full Screen Edit feature, you tell WYLBUR to:

• SET WINDOW	to alter one or more of the default window options until further notice. Options include: FORMAT (STANDARD, CARD, LONG), DISPLAY (CONTEXT, DENSE), FILLER (NULL, BLANK) and PAGE SIZE.
• SHOW WINDOW	to display the current default WINDOW OPTIONS.
• WINDOW UP/DOWN • PAGE BACK/FORWARD	these synonomous commands shift the display backward or forward one page.
• WINDOW LEFT/RIGHT	shift the window sideways, one page.
• WINDOW COLUMN	specifies the text column in which you want the display to begin.
• REVIEW	the ACTIVE file in window format. Similar to LIST, ACTIVE file lines are protected.
• UPDATE	the ACTIVE file using Window Mode FSE.
• DEFINE	a value for up to 24 Program Function Keys. Provision is made for the user to allow for a variable to be substituted into the expanded command before it is passed to WYLBUR for execution.
• UNPROTECT	the current screen so you can edit it. This is the most useful when you are REVIEWing and spot something that needs fixing.
• COLLECT	lines in your ACTIVE file by displaying empty lines for you to type into. ENTER is pressed only once per page of lines and unused lines are discarded before the screen is re-displayed.
• INSERT	single lines between existing lines while displaying surrounding contextual lines.

• DELETE	line from the ACTIVE file and re-display the window.
• CHANGE	character strings or specific text columns to a specified string and re-display the window.
• COPY	lines from one place in the ACTIVE file or from an external dataset to a specified place in the ACTIVE file and re-display the window showing the lines in their new location.
• MOVE	lines from one place in the ACTIVE file to another and re-display the window showing the lines in their new location.

REMOTE JOB MANAGEMENT

Remote job management lets you submit batch jobs from a terminal connected to the central computer. You can track the status of jobs from the terminal, and you can fetch job output and review it at the terminal

You use your ACTIVE file as a work area for composing your job -- JCL, source code, and input data if necessary. The job can be entered into the ACTIVE file with the COLLECT command, or built by using one or more permanent files. Then, using a WYLBUR command, you submit your job to the central computer. Any job you can submit through conventional batch methods can be submitted remotely with WYLBUR.

You need no special JCL, no JCL changes in existing jobs or extra job steps to submit the job and fetch it remotely. In operating systems utilizing JES1 or JES3, it may be more expedient to modify your SYSOUT DD statements to refer to a special SYSOUT Class in order to retrieve job output at the terminal. WYLBUR can assign jobnames for you that consist of your USERID and a suffix. WYLBUR constructs jobnames in this manner to help identify job ownership.

When you want to look at job output, you use a WYLBUR Command to retrieve the output into your ACTIVE file. All WYLBUR text handling commands can then be used to examine the output. You may FETCH all or part of your job output as many times as you want. You may also direct output to other devices for printing or remove (PURGE) it from the system

To use the remote job management facility, you tell WYLBUR to:

• RUN	a job by submitting all or part of the ACTIVE file to the computer's input queue.
• LOCATE	one or more jobs and display their status at the terminal.
• FETCH	all or part of a job's output into the ACTIVE file for examination.
• PRINT	the output of a job on a high speed printer, and thus release it from the system.
• PURGE	a job's output when it is not needed, instead of printing it.
• CANCEL	executing jobs and remove them from the system.
• HOLD	a job by having it placed in hold status.
• ROUTE	the output of a job to a particular output destination.
• RELEASE	a job that was placed in a hold status.

Example of Remote Job Management

This example shows the process involved in creating JCL for a job and using WYLBUR to submit it. The job will be created, run, and fetched for examination.

```
COMMAND? collect
   1. ? //        JOB
   2. ? //       EXEC PGM=IEFBR14
   3. ? //SYSPRINT DD  SYSOUT=A
   4. ? //DD1      DD DSN=TEST.DATALIB,DISP=(NEW,CATLG),
   5. ? //            UNIT=3330,DCB=(RECFM=U,LRECL=3156,
   6. ? //            BLKSIZE=3156,DSORG=PO),VOL=SER=WYL001,
   7. ? //            SPACE=(TRK,(19,,9))
   8. ? (ATTN) ***
COMMAND? run class=s fetch
JOB 1073 OZUSR053 SUBMITTED
COMMAND? locate
JOB 1073 OZUSR053 AW FETCH
COMMAND? fetch *
OK TO CLEAR? OK
COMMAND? list 'cond code'
  10.     IEF142I OZUSR053 - STEP WAS EXECUTED - COND CODE 0000
COMMAND? purge *
OK
```

Explanation:

The COLLECT command is issued in order to build JCL for a job to be run.

The RUN command tells WYLBUR to submit the ACTIVE file to the central computer's input queue. The 'FETCH' parameter on the RUN command tells WYLBUR to hold the job output for retrieval at the terminal.

After the RUN command is issued, WYLBUR shows you the jobname and number. A LOCATE command causes WYLBUR to locate all your jobs in the system, and display their status. When a job is 'AW FETCH' it has been run and may be retrieved at the terminal.

The FETCH command retrieves job output into your ACTIVE file. '*' refers to the last job you submitted. FETCH without additional parameters tells WYLBUR to bring in all the output for the job, although you can select JCL, job log, or dd statement output individually by issuing other parameters.

Job output is brought into the ACTIVE file. Since the job input was still there, WYLBUR asked for permission to clear the ACTIVE file.

Once job output is in your ACTIVE file, you can look at all of it with a LIST or REVIEW command, or you can list only the important lines by telling WYLBUR to list lines containing certain strings.

Fetching job output does not destroy it. You may print or purge it as necessary.

UTILITY FUNCTIONS

When you save or scratch disk datasets, update the system catalog, or compress partitioned data sets, you are performing utility functions. WYLBUR allows you to do these functions on-line by simply issuing WYLBUR commands Because WYLBUR is a privileged program, it can directly access and modify the system catalog, show you entries in a PDS directory, list any or all of the dataset names on a particular volume, as well as many other timesaving functions. You save time by performing these functions on-line rather than running a series of utilities to get or change the same information.

WYLBUR dataset names usually conform to the standard structure shown below Naming conventions can be customized for special requirements. For a detailed look at installation controlled variables, refer to the "CUSTOMIZING" section of this manual.

```
    WYL.(group).(initials).(prefix)

                          dataset prefix
                 user identification
       group identification
    standard WYLBUR high-level index
```

WYLBUR dataset names are qualified in this manner for a number of reasons:

1. All WYLBUR datasets are easily identified by the index 'WYL'. WYLBUR can also find its data sets faster when the 'WYL' datasets are listed in a CVOL connected to the system catalog. This helps reduce response time and overhead.

2. All WYLBUR datasets belonging to a particular group are identified by the group qualifier. That makes it easy for management or the operations staff to monitor disk usage.

3. All WYLBUR datasets belonging to a particular user can be identified by the user's initials.

4. The fourth qualifier, prefix, allows you to select a descriptive name for your datasets in addition to the WYLBUR index, group, and user qualifiers.

When datasets are allocated in this manner, you can let WYLBUR supply the qualifiers when accessing datasets. Instead of typing fully-qualified dataset names in WYLBUR utility function commands, you need only specify a prefix or member name. WYLBUR builds the rest of the qualifiers using the high-level index, signed-on group, and initials Of course, any or all of the qualifiers can be overridden if you should want to access some other user or group's datasets, or non-WYLBUR datasets. Using WYLBUR conventions for dataset names lets you type less which in turn makes your terminal session more productive.

What Are the Utility Function Commands?

To manage datasets on disk, you tell WYLBUR to:

• SAVE	your ACTIVE file as a disk dataset or PDS member.
• SCRATCH	datasets or members by removing them from disk storage.
• USE	a dataset by bringing a copy of it into your ACTIVE file for examination or editing purposes.
• CATLG	datasets by adding them to the system catalog.
• UNCATLG	datasets by removing them from the system catalog.
• RECATLG	datasets by changing their volume pointer in the system catalog.
• CONDENSE	a PDS by running a compress utility.

To look at the catalog, dataset statistics (VTOC information) or PDS directories you tell WYLBUR to SHOW:

• CATALOG	to examine the system catalog for particular entries.
• DSNAMES	to list all the datasets and statistics on a certain volume or only those datasets that conform to a certain pattern.
• DIRECTORY	to list any portion or all of the entries in a PDS directory.
• SPACE	to determine the amount of free space on a volume, or the space limits and current status for a given user.

Example of Utility Functions.

This example shows a sign-on procedure, and dataset members being SAVEd and SCRATCHed:

```
OBS WYLBUR/MVS REL 5.1 TERMINAL 15:28:48 05/01/80

USER ID? ozusr
KEYWORD?
ACCTNO? 7309b00
ACCTNO IN EFFECT: 7309b00000
COMMAND? show dir
WYL.OZ.USR.LIB
COMPILE
DEMEXEC2
DEMOEXEC
EXEC2
EXEC3
LOGON
COMMAND? use #compile
COMMAND? cha 'member' to 'mypgm'
    2.      //JS010 EXEC WUNPRESS,G=OZ,I=USR,N=MYPGM
    6.       NAME    MYPGM(R)
COMMAND? save #compil2
MEMBER COMPIL2 SAVED IN WYL.OZ.USR.LIB
COMMAND? scratch #compile
MEMBER COMPILE SCRATCHED FROM WYL.OZ.USR.LIB
COMMAND? show dsn like $obsa.cost on cat all
USER12
OBSA.COST -- 1 of 19 TRKS -- IS, FB/24/3240/4 -- CR:  08/11/76
COMMAND? show catlg for $obsa.
OBSA
AUDIT - ON SYS504
BILLJES - ON USER12
B158 - INDEX
CDETAIL - ON SYS504
COST - ON USER12
DETAIL - ON SYS504
DETAILWK - GDG 6 GENS
DRECYCLE - ON SYS504
DSMFTLMS - ON SYS504
DTAPESUM - GDG 3 OF 1
COMMAND? sho cat for $obsa.detailwk
OBSA.DETAILWK
G0102V00 - ON 004736
G0101V00 - ON 002988
G0100V00 - ON 000517
G0099V00 - ON 004214
G0098V00 - ON 004305
G0097V00 - ON 006245
COMMAND?
```

Explanation:

The absense of a dataset name on the SHOW DIR command tells WYLBUR to build the name using the standard defaults. The dsname constructed by WYLBUR was therefore: WYL.OZ.USR.LIB

WYLBUR lists the dataset name followed by all the members in the library.

A copy of the member COMPILE is brought into the ACTIVE file.

After a global change, the ACTIVE file is saved in the user's library as a new member called COMPIL2. COMPILE is then scratched from the user's library.

The dollar sign ($) identifies a non-WYLBUR dataset in the SHOW DSN command. The 'ALL' parameter tells WYLBUR to list the dataset characteristics.

The user then requests System Catalog information for all non-WYLBUR datasets beginning with the OBSA. index.

WORD PROCESSING-DOCUMENT FORMATTING

WYLBUR is an excellent vehicle for word processing and document formatting applications It can be effectively used either by secretarial personnel for general correspondence, form letters, proposals, contracts, etc. or by publications personnel for creation and update of user documentation, technical manuals and other large documents.

WYLBUR offers text editing capabilities utilizing medium to large scale general purpose IBM computers. Utilization of these large mainframe computers provides advantages not available in special purpose minicomputer Word Processing systems. Some of these advantages include:

- Storage capacity - If you are concerned with editing large files such as textbooks, rate tariffs or manuals, WYLBUR and your computer can handle fast storage and retrieval for these types of files. The power of a large scale computer is also important when scanning large files or making global changes.
- Cost Savings - If you have WYLBUR, you need not worry about the expense or reliability of additional hardware except, of course, the terminals you intend to add.
- Flexibility - By using WYLBUR's powerful Execute Programming capability, installations can customize Word Processing functions based on their own unique requirements.

Although WYLBUR has capabilities for both document formatting and word processing, these two applications are considered separately because they are accomplished using different facilities.

Word Processing

Definition: Capability for creation, easy correction and high quality printout of smaller documents (letters, proposals, white papers, contracts, etc.). These tasks are normally accomplished by secretarial personnel who have received training in necessary techniques The reader is referred to the OBS WYLBUR Reference Manual for a detailed description of the commands mentioned below.

Approach

Documents are created from source by typing (using the WYLBUR COLLECT Command) into the WYLBUR Active file and creating a compressed OS dataset or Library member (SAVE Command) containing the text. Each line in the file is numbered automatically by WYLBUR and subsequent editing commands may refer either to line number or text content. The document is then available for immediate access (USE command) and correction (UPDATE command) using Full Screen Edit on 3270-compatible terminals. After original creation, documents are printed at a high quality printing terminal (LIST command) for review by author. Corrections require re-typing only of the items in error and the document can easily be re-listed after corrections.

Hardware

Minimum requirement would be one 3270-compatible terminal (with upper/lower case feature) for creation/editing and one high quality output terminal (E.G. DIABLO 1620) for printing. Ideally, each secretary would have a 3270 terminal and one output terminal for each three 3270 terminals. Although some installations may prefer exclusive use of start-stop terminals, Full Screen Edit is available only on 3270 compatible devices.

Functions

Capabilities inherent in WYLBUR for Word Processing are all on-line functions including the following:

Text creation - COLLECT command

Editing - Full Screen Editing via UPDATE command or EDIT and MODIFY commands for start-stop terminals

Global Changes - CHANGE command

Paragraph or Line Insertion - INSERT and COLLECT command

Paragraph or Line Deletion - DELETE command

Justification of Paragraphs - JUSTIFY command provides left and right hand justification of paragraphs. Any line width may be specified, with or without paragraph indentation. Most often used to re-justify paragraphs after corrections have been applied to text. The ALIGN command provides ragged right justification.

Centering - CENTER command

Pagination - The MARKER option of the LIST command provides pagination capabilities. When LISTing at the output terminal, the terminal will pause whenever the MARKER is encountered in column 1 to allow a fresh page to be fed in.

Moving Blocks of Text - MOVE command moves lines or paragraphs to different locations

Duplicating Text From Other Documents - COPY command with FROM option

Example of Word Processing with WYLBUR

```
COMMAND? collect
    1.    ? Getting To Know WYLBUR
    2.    ?
    3.    ? If you are using WYLBUR for the first time
    4.    ? it will be helpful for you to first read the User Guide.
    5.    ? Basic Editing Commands you will want to know about are:
    6.    ?
    7.    ? CHANGE COLLECT COPY DELETE INSERT MOVE
    8.    ?
    9.    ? If you have access to 3270 terminals, you will want to
   10.    ? learn the Full Screen Editing Commands including:
   11.    ?
   12.    ? REVIEW PF PB UNPROTECT UPDATE WL WR WC
   13.    ?
   14.    ? After you have read about these commands, LOGON to a
   15.    ? terminal and experiment with their use.
   16.    ? (ATTN)***
 COMMAND? set length 60
 COMMAND? center 1
 COMMAND? align 3/5 9/10 14/15 indent 0 4 even
 COMMAND? align 7 len 8 even
 COMMAND? justify 12 len 40 indent 25 0 even
 COMMAND? list
  1.                      Getting to Know WYLBUR
  2.
  3.           If you are using WYLBUR for the first time it will be
  4.       helpful for you to first read the User Guide.  Basic Editing
  5.       Commands you will want to know about are:
  6.
  7.       CHANGE
  7.001    COLLECT
  7.002    COPY
  7.003    DELETE
  7.004    INSERT
  7.005    MOVE
  8.
  9.           If you have access to 3270 terminals, you will want to
 10.       learn the Full Screen Editing Commands including:
 11.
 12.                             REVIEW   PF   PB
 12.001                          UNPROTECT
 12.002                          UPDATE WL WR WC
 13.
 14.           After you have read about these commands, LOGON to a
 15.       terminal and experiment with their use.
 COMMAND? cha '      ' 1 to '' in 12/12.002
 12.                         REVIEW   PF   PB
 12.001                      UNPROTECT
 12.002                      UPDATE WL WR WC
 COMMAND? 16 %
 COMMAND? list
```

```
  1.                        Getting To Know WYLBUR
  2.
  3.          If you are using WYLBUR for the first time it will be
  4.      helpful for you to first read the User Guide.  Basic Editing
  5.      Commands you will want to know about are:
  6.
  7.      CHANGE
  7.001   COLLECT
  7.002   COPY
  7.003   DELETE
  7.004   INSERT
  7.005   MOVE
  8.
  9.          If you have access to 3270 terminals, you will want to
 10.      learn the Full Screen Editing Commands including:
 11.
 12.                              REVIEW  PF   PB
 12.001                          UNPROTECT
 12.002                          UPDATE WL  WR  WC
 13.
 14.          After you have read about these commands, LOGON to a
 15.      terminal and experiment with their use.
 16.      %
 COMMAND? list off marker %
 JOB 0865 OZUSR544 SUBMITTED
 COMMAND?
```

Explanation

In COLLECT mode, the operator enters text without concern for line length or word placement.

The value of LENGTH is then set to 60. Subsequent formatting commands will thereafter assume that 60 is the desired line length unless a different length is requested.

The title line is CENTERED.

Three different groups of lines are then ALIGNED so that no line will exceed 60 characters, yet each line will contain as many characters as possible The INDENT parameter tells WYLBUR to indent each line 0 characters except for the first line of each paragraph which is to be indented 4 spaces New paragraphs are assumed following each blank line. EVEN tells WYLBUR that the user wants to change existing indentation

The command words are then ALIGNED based on a length of 8 to cause each command to print on a separate line

The FSE commands are JUSTIFIED so that left and right margins are exactly even (if possible) yet no line is to exceed 40 characters in length

The file is then LISTed for review

The user decides to center the FSE command group by deleting 4 blanks to the left of each group with the CHANGE command. The CENTER command would have destroyed the justification of the groups.

The line number command is then used to create line 16 containing a % in position 1. The % is subsequently used as a Marker to cause a page eject at the high speed printer, via the LIST OFFLINE Command or the marker may also be used to cause a typewriter terminal to pause when the marker is encountered thus allowing a fresh sheet of paper to be fed in.

The actual syntax of the various commands need not be learned by users if the installation desires to customize Word Processing capabilities through the use of WYLBUR EXECUTE Programming and the WYLBUR DEFINE command. These functions provide the capability for the installation to write customized on-line programs which will prompt users for desired displays and utilize menu functions to simplify use. The DEFINE command also allows complete utilization of 3270 Program Function Keys so users can be taught to press a particular PFK for a desired function rather than learning commands.

Document Formatting

Definition: Capability for creation, easy correction and high quality printout of large documents such as user or technical manuals. These tasks are normally accomplished by personnel with some technical ability to utilize document formatting commands which are imbedded in the text and processed by a document formatter which is invoked either on-line or via batch submission.

Approach

Documents are created in the same manner as described for the Word Processing function. In this case however, Document Formatting Control Words are imbedded in the text file. These Control Words are processed by the Formatter and the result is a finished document.

Two different Formatters are recommended for use with WYLBUR. The first is PAGER, which is supplied by OBS and included with a WYLBUR license. PAGER is a series of EXECUTE Programs which process Control Words in a text file which is contained in the WYLBUR Active File. PAGER provides formatting capabilities including:

- Automatic Pagination and Page Numbering
- Control of Page Headings and Footings
- Generation of a Table of Contents or Index

The second recommended formatter is University of Waterloo SCRIPT available for a nominal licensing fee and supported by UOW. SCRIPT provides capabilities for any function that could reasonably be expected from a document formatter. It operates as a batch program but works hand-in-glove with WYLBUR. WYLBUR is used as the text editing vehicle for creation of original text, insertion of SCRIPT Control Words, correction of errors in text and errors in SCRIPT Control Words, submission of SCRIPT formatting runs to batch, and review of formatted output at the terminal prior to printing.

The functions of UOW SCRIPT are far too numerous to list here but are generally sufficient to meet any document formatting need. Functions available include:

- Pagination
- Table of Contents and/or Index Creation
- Automatic Justification
- Footnotes
- Page Headings and Footings
- Hyphenation Dictionary

- Subscripts and Superscripts
- Underscore
- Macro Language

It should be noted that although SCRIPT runs in batch, its utilization is only effective when combined with a text editor and RJE package such as WYLBUR.

Example of Document Formatting with PAGER

In the following example, a document is shown prior to being formatted by PAGER. PAGER Commands are interspersed in the text. A $ or # in column one identifies a PAGER Command. The following PAGER Commands are used in the example:

$HEADROUT	Specifies the name of the Execute Program which will be called by PAGER after each page is processed. The Heading Routine controls page numbering, index or table of contents generation and page headings or footings.
$PAGESIZ	Tells PAGER the maximum number of lines to be printed on each page. This value remains in effect for the entire document unless overridden by the $NEWPAGE statement.
$LENGTH	Causes a warning message to be issued if a text line is modified to exceed the specified LENGTH.
$TITLE	Causes a page break and passes the value of the provided title string to the Heading Routine for processing.
$SECTION	Causes a page break and passes the value of the provided section string to the Heading Routine for processing.
$TOPIC	Causes a page break and passes the value of the provided topic string to the Heading Routine for processing.
$NEWPAGE	Forces a page break and call to the Heading Routine.
#TCON	Causes a Table of Contents entry to be generated without forcing a page break.

A standard Heading Routine is provided with PAGER. This routine will number pages, print title, section and topic headers, and build a Table of Contents.

The following is an example of an Input Text File, containing PAGER

Control Statements.

```
COMMAND? list
  0.001    $HEADROUT WYL.OZ.USR.EXEC#UHEAD ON WYL101
  0.002    $PAGESIZ 15
  0.003    $LENGTH 80
  0.004    $TITLE OBS UTILITIES MANUAL
  0.005    $SECT BATCH UTILITIES CREV I.CP'
  2.
  3        OBUPDTE is a driver program that invokes the IBM utility
  4.       program, IEBUPDTE.
  5.
  6.       OBUPDTE allows concurrently running jobs to share the same
  7.       existing output data sets by insuring that exclusive
  8.       control is given during each updating process.
  9.
 10.       OBUPDTE should always be used instead of IEBUPDTE, for all
 11.       processing functions.
 12.
 13.       All functions are identically controlled by IEBUPDTE
 14.       utility control statements.
 16.       #TCON EXAMPLE
 17.       JCL EXAMPLE:
 18.
 19        //STEP1     EXEC PGM=OBUPDTE
 20        //SYSPRINT DD    SYSOUT=A
 21.       //SYSUT1    DD    DISP=SHR,DSN=XXX.PROCLIB
 22.       //SYSUT2    DD    DISP=SHR,DSN=XXX.PROCLIB
 23.       //SYSIN     DD    DATA
 24.       ./  ADD  NAME=NEWPROC,LIST=ALL
 25.       ./  NUMBER  NEW1=1000,INCR=1000
 26.       //NEWPROC   EXEC PGM=PGMNAME
 27.       //........ DD    ........
 28.       //........ DD    ........
           /*
 COMMAND? execute from pager on catlg clear
 TC OFFSET IS 100
 DONE
 COMMAND? list offline unn upper cc
```

The following sample represents the output that was produced after the Table of Contents has been moved to front of the document. Actual page ejects are represented by four blank lines in the sample.

EXECUTE PROGRAMMING

One of the most versatile facilities of WYLBUR is its Interactive Programming Language (EXECUTE). EXECUTE programming provides a high-level interactive programming language that is available to every user. High-level means that it is simple to use. Interactive means that you control what it does -- you can have it converse with you at the terminal, prompt for information, and display results on-line. There are several sophisticated general arithmetic and logical functions available that you can use in an EXECUTE program to make it as powerful and effective as any other high-level programming language.

An EXECUTE program is a collection of WYLBUR commands executed in a controlled sequence. EXECUTE programs may be executed from the ACTIVE file or from a disk dataset. There are commands for branching, Branch-And-Link, and transferring control to another EXECUTE program. You may use string, decimal, or integer variables as well as many other WYLBUR special keyword variables. You can instruct WYLBUR to prompt the terminal user for information and give your program the information that is typed in. You can have WYLBUR substitute values at certain points in your program or evaluate an expression and then substitute the value of the expression. There are also decision-making commands. Once you have composed your EXECUTE Program in your ACTIVE file, you tell WYLBUR to execute it. WYLBUR takes the program from the ACTIVE file and puts it into your secondary work area called the EXECUTE File. EXECUTE programs are interpretive in nature. WYLBUR processes each command, resolving expressions and making substitutions, if specified. If there are errors in the program, WYLBUR interrupts the command that failed so you can correct it on the spot and resume execution. When you are satisfied with the program and want to keep it for future use, you can save it on disk.

There are many applications for EXECUTE programs. Any time you need input from a user in order to set up documents or run jobs, you can use an EXECUTE program to automate the process. Building customer lists? Writing letters? Performing operating system maintenance? Translating programs from one operating system to another? Let an EXECUTE program do the work for you.

What are the EXECUTE Programming language commands?

There are some special WYLBUR commands that you can use as program instructions in an EXECUTE Program. Additionally, you can use any of the other text entry and editing, document processing or remote job processing commands already described in the preceding sections. The special EXECUTE-oriented commands are:

- EXECUTE To initiate an EXECUTE Program, or perform branching within it

• SET VALUE	to assign a value to any of the string, decimal, integer or keyword variables.
• READ	to get input from the user at the terminal or from the ACTIVE file.
• IF	to make a decision.
• RESTORE	to bring an EXECUTE Program out of the EXECUTE File into the ACTIVE file for correction when debugging.

To use the features of the WYLBUR Preprocessor which performs expression resolution and value substitution, you will want to use special global parameters by telling WYLBUR to SET or SHOW:

• ESCAPE	the character that triggers text substitution in an instruction.
• SKIP	the character that causes substitution to be bypassed on certain words.
• RESCAN	a count of how many times to process an EXECUTE program when nested levels of substitution are required.
• EXECUTE	a 'trace' facility that determines if EXECUTE instructions will be logged to the terminal as they are executed.
• RETURN	the 'branch-and-link' line number.

Briefly, the general arithmetic and logical functions available to you through the EXECUTE programming language are:

String (character) Functions:

• SUBSTR	produce a substring of a string.
• INDEX	find the starting column of a particular string.
• VERIFY	examines two strings to determine whether or not the first string is contained in the second.
• FIND	examines two strings to determine the position of the first character in the first string that is found in the second string.
• SIZE	determine the length of a string

Encode/Decode Functions

• DISPLAY	converts arguments to internal hexadecimal representation.
• SHEX	converts hexadecimal to EBCDIC.
• NHEX	converts hexadecimal to integer.
• WHEX	converts hexadecimal to decimal.

Conversion Functions:

• NCONVERT	converts string or decimal to integer.
• WCONVERT	converts string or integer to decimal.
• SCONVERT	converts integer or decimal to string.

Expression Evaluation:

+	Add
-	Subtract
*	Multiply
/	Divide
( )	Parenthesis to alter order of evaluation
||	Concatentation

Relational Operators:

EQ	equal
NE	not equal
LT	less than
LE	less than or equal
GT	greater than
GE	greater than or equal

EXECUTE Program Example:

```
 1.   SET EXEC NOLOG TERSE
 2.   SET ESCAPE &
 3.   READ STRING S1 PROMPT 'STARTING MBR? '
 4.   IF (S1 EQ '') EXEC 3
 5.   READ STRING S2 PROMPT 'ENDING MBR? '
 6.   IF (S2 EQ '') EXEC 5
 7.   SET VALUE W0=59000
 8.   COLLECT DIR FOR $SYS1.PROCLIB FRO &S1 THRU &S2 ON CAT
 9.   DEL F
10.   NUM 59001
11.   SET VALUE W0=W0+1
12.   SET VALUE W1=LAST
13.   READ STRING S5 USING &W0
14.   COPY F/L FROM $SYS1.PROCLIB#&S5 PLUS 0 ON CAT
15.   COMMENT   MEMBER &S5
16    MOD ' EXEC ' AND ¬ '*' 3
17.   SAV $SYS1.PROCLIB#&S5 LINES F/58999 ON CAT REP
18    SET VALUE W0=W0+1
19.   IF (W0 GT W1) EXEC 22
20.   DEL F/58999
21.   EXEC 13
22.   SET ESCAPE
23.   COMM END
```

Explanation:

This EXECUTE program will search each member of PROCLIB, starting and ending with member names or member name patterns specified by the user. For each applicable member, the Program will find all EXEC statements, allow the user to modify them and replace the updated member in PROCLIB.

NOLOG TERSE is specified to prevent WYLBUR from logging each command as it is executed.

The Escape character is specified which tells WYLBUR to perform substitution when the escape character is encountered

The starting and ending Member names are asked for. Note that if no response is provided, the request is repeated.

The applicable portion of the directory from PROCLIB is stored in the ACTIVE file, beginning at line number 59001.

Counters are initialized and the first member name is read into string variable 5 (S5).

The actual member is then read into the beginning of the ACTIVE file and a line is typed at the terminal informing the user of the applicable member name.

The MODIFY command is issued so that each JCL EXEC statement that is not a comment may be modified by the user. The updated member is then replaced in PROCLIB

If more members are to be processed, the current member is deleted from the ACTIVE file and processing continues at line 13.

Notes

Depending on the complexity of the required modifications, it may have been feasible to code the EXECUTE program so that it actually made the modifications, rather than having the user be responsible for them. The STRING Processing Functions provide powerful facilities for this type of task.

The Branching function within EXECUTE programs utilize WYLBUR line numbers. An EXECUTE Program Assembler is provided with WYLBUR so labels may be issued in lieu of actual line numbers.

EXECUTE Programs Distributed with WYLBUR

A number of EXECUTE Programs are provided with your license to WYLBUR. Some of these programs will be useful to your installation on a daily basis, some will provide excellent examples of EXECUTE Programming techniques and some are just plain fun. Most of these programs will require some minor customization by the installation for such things as data set names and disk volume serial numbers. Documentation and/or "EXEC ASSEMBLER" source is provided for applicable EXECUTE Programs in a separate library which also is provided on your distribution tape. Provided programs include:

ASSEMBLE	This "EXEC ASSEMBLER" allows the use of labels when writing EXECUTE Programs in lieu of referring to actual line numbers. This can be especially helpful when you are writing a fairly large EXECUTE Program which may need to be renumbered several times. The Assembler also optimizes your EXECUTE Programs by compressing unnecessary characters and provides a symbolic notation for pre-defined variables.
HELP	Displays syntax and detailed command description for requested command. ACTIVE file remains undisturbed.
LIBALLO	Builds JCL and submits a batch job to allocate a PDS (Library) for the user.
LIBEXP	Builds JCL and submits a batch job to expand the size of a PDS.
PAGER	The OBS WYLBUR Document Formatting EXECUTE Program.
RANDOM	A randomizing routine which may be invoked by other EXECUTE Programs
SQRT	Provides the square root of a user supplied number.
SHOWVT	Displays disk volume usage by the WYLBUR system (for System Programmers use - requires minor customizing by the installation).
TRKCAP	Displays track utilization based on user specified LRECL, Key length and Block size.
GAMES	A number of popular games are provided for your enjoyment.

THE OPERATIONS SUPPORT SYSTEM

INTRODUCTION

The OBS Operations Support System (OSS) was developed by OBS because we felt there must be a better way to run a data center. The major goals of the system were:

- To develop uniform, easy to update, reliable, and complete documentation for all production jobs run at OBS.
- To provide as much information about the jobs as necessary to allow operations personnel to make informed decisions when prioritization of work becomes critical.
- To develop audit trails to facilitate research if an output item is not received by the customer
- To automate, to whatever extent possible, the job scheduling, submission and restart process to minimize card handling errors and errors normally caused by erroneous assumptions that can often be made by operating personnel.

Since WYLBUR was already being used for program development at OBS, and it provided most of the facilities that seemed necessary, it was chosen as the vehicle for implementation of OSS. OSS takes advantage of WYLBUR'S capabilities for:

On-line interaction with a user (Execute Programs)

Global search and retrieval

Job submission

General text creation and manipulation

File Description

Date (Calendar) File - A WYLBUR file which contains one entry (line) for each day on the calendar for the next 18 months. Each date is given scheduling attributes, E.G., 010278 is a Monday, first Monday of the month, a weekday, Julian date 78002, etc

Scheduling Requirements File - A WYLBUR file which contains two types of entries. The first type is a "Batch" entry which indicates a group of jobs which are logically associated and normally run in a pre-defined sequence. The second type is a "Job" entry which describes a single job. Both types of entries contain information describing the types of days when the job or batch should be scheduled to run, i.e., weekdays, Saturdays, first Monday of the month, etc.

Job Profile Library - A large "WYLBUR" library which contains many members of two different types.

Batch Expansion Members - This type of member contains a list of jobnames associated with a "Batch" as described above. The member name for this type of member is equal to the "Batch" name.

Job Profile Members - Each job profile member (member name is equal to jobname) contains the following groups of information:

> Operator Recap Information - Information concerning job execution requirements which would be important to the computer operator. For example, normal run time, other job dependencies, maximum tape drives, number of scratch tapes needed, input tape files, comments, etc.
>
> Production Control Information - Information important to proper job output handling. For example, number and type of printed outputs, decollating and bursting information, any output tapes, non-standard completion codes, pick up or distribution information, job sensitivity information, etc.
>
> Restart/Rerun Information - Special instructions to be followed in case the job must be rerun or restarted.
>
> Job Submission Information - This consists of the production JCL required to run the job.

Functional Description

Function - To schedule and submit the production jobs for a given day. Refer to Figure 1 for schematic representation.

1. Scheduler invokes the EXECUTE Program which begins the process and provides the calendar date which is to be scheduled. The EXECUTE Program finds the proper entry in the Calendar File, stores the "attributes" for that date, and continues processing.

2. The Scheduling Requirements File is searched for every job or "batch" which is to run on any day and has an "attribute" equal to those associated with the given date. We now have a list of batches and jobs which are to be run on the scheduled date. The process may be interrupted at this stage to add or delete batches or jobs which have special requests pending for today's work.

3. For each Batch, the Batch Expansion Member is found by the EXECUTE Program. At the end of this process, we have a job list which contains a single entry for each job to be run on the scheduled date.

4. The scheduler may again wish to manipulate the job list to insert job names of On-Request jobs or One-Time jobs and delete any entries for jobs which are not to run because of special requests The job submission process may be suspended at this point to ensure that all special requests have been received and acted upon

5 When job submission is desired, the scheduler invokes the job submission EXECUTE Program which then reads the job list. For each entry in the job list, the Job Profile Library member is found and the Job Submission Information (JCL) is captured. In this manner, one big file of run-time JCL is built. Once this is accomplished the scheduler may wish to insert special overrides which may have been requested for certain jobs for this run only.

6 With one command, the production workload is submitted into the input queues awaiting release by the operator.

7 A high priority batch job is submitted to produce Recap reports for the operator and production control (PC). The report program reads the job list created in Step 4. For each entry in the job list, the Job Profile Library member is found and the appropriate Operator and PC Recap information is extracted and formatted into two reports The system catalog is read at this time to determine volume serial numbers for input data sets and a tape pull list is also created.

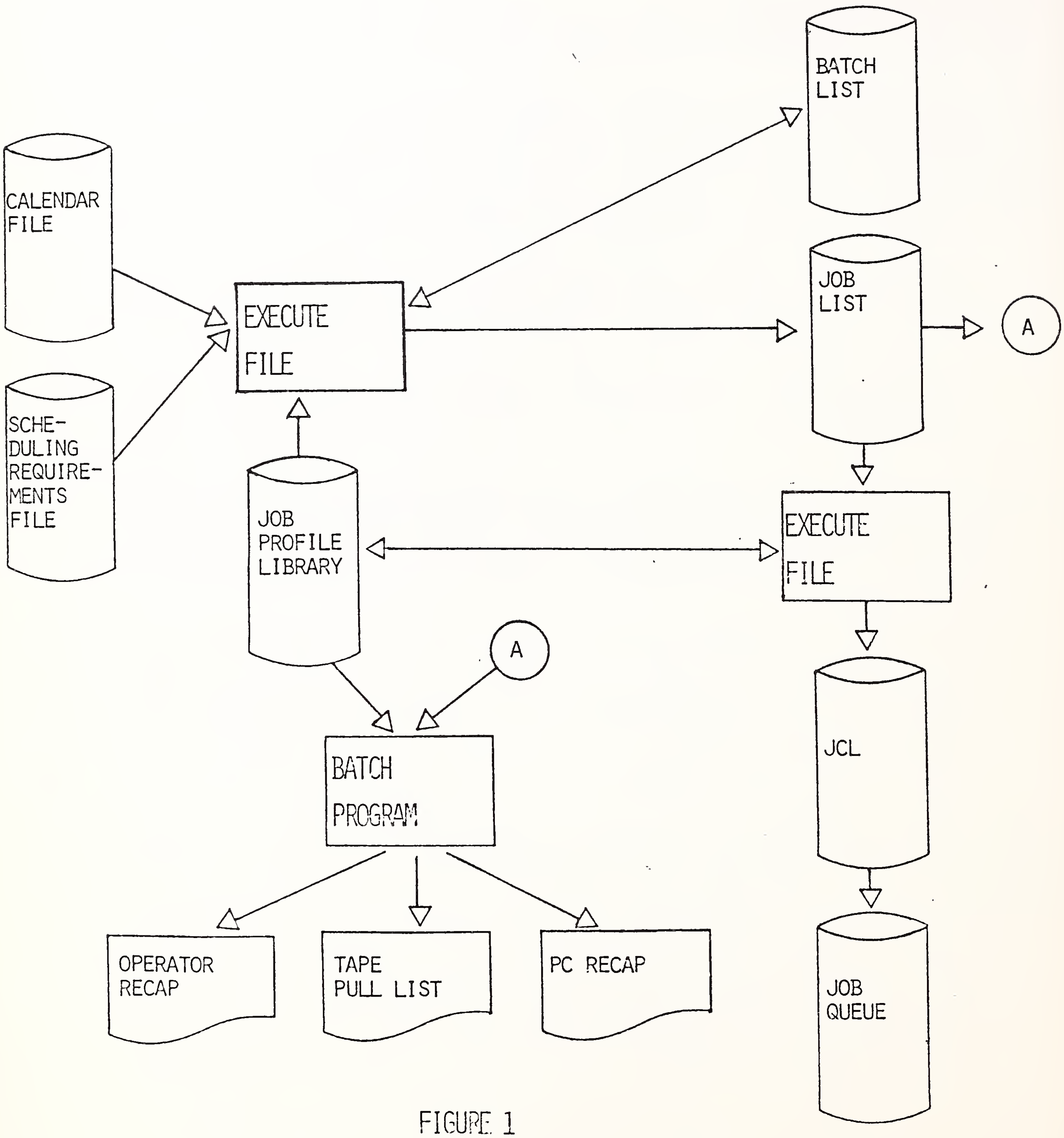

BATCH LIST
CALENDAR FILE
JOB LIST
A
EXECUTE FILE
SCHE-DULING REQUIRE-MENTS FILE
JOB PROFILE LIBRARY
EXECUTE FILE
A
BATCH PROGRAM
JCL
OPERATOR RECAP
TAPE PULL LIST
PC RECAP
JOB QUEUE

FIGURE 1

8. At this point, the following work has been accomplished:

- All production jobs have been submitted into the proper job class and await release by the operator.

- The operator has a report which contains all execution requirements for _those_ _and_ _only_ _those_ jobs which are to run tonight.

- A tape pull list has been created so that the majority of input tapes can be pulled and in the computer room prior to execution of the jobs.

- The production control department has a report describing which output will be produced tonight, how it is to be handled (bursted, decollated, boxed, labeled, etc.), its distribution and timing requirements, an indication of which jobs have the highest priority in case some work must remain incomplete and documentation for acceptable condition codes for each job step which allows personnel to check for successful completion of each job.

9. The operator and production control personnel now annotate their reports as work is completed. This effectively provides a checklist approach to ensure that each job is completed accurately and in its entirety.

10. If there is a system crash or job failure during production processing, affected jobs must be rerun or restarted. In order to accomplish this, the shift supervisor or designee will use WYLBUR's facilities to clean up any duration-of-job disk datasets and do the necessary catalog maintenance. The supervisor then invokes the Rerun/Restart _Execute_ _File_ and is asked to key in the jobname(s) of the appropriate job. The _Execute_ _File_ then searches the Job Profile Library and locates the Rerun/Restart information for the job. This information is displayed for the supervisor to read. The run JCL is then left in the active file for the supervisor to insert overrides if necessary and resubmit the job. An entry is automatically inserted in a "trouble log" dataset which is then reviewed each morning by data center management for a quick overview of any problems which occurred the previous night.

<u>Benefits</u>

1. Probably the most important benefit of OSS is the single source, easily updated, machine readable, job documentation library. Through on-line access via WYLBUR, job documentation is easy to update. It is easily reproduced in whole or in part. With single source, on-line accessable documentation it is no longer necessary to try to keep several copies of run books up to date.

2 Formatted documentation is available to both operations and production control personnel. Neither group needs to contend with documentation pertinent only to the other group. Each group is presented with a fresh copy <u>each</u> <u>day</u> of the documentation applicable to those jobs which are to be run that day.

3. Card handling errors are eliminated. No more errors because of Read Checks, Feed Checks, card jams, dropped decks or forgotten JCL override cards. Your shop becomes a lot cleaner and neater to help facilitate an organized operation.

4. Reruns and restarts are less apt to cause problems because personnel are forced to review specific restart procedures for each job. WYLBUR itself helps simplify this procedure by providing on-line utility functions such as SCRATCH and UNCATALOG.

5. With the scheduling information readily available in machine readable format, the data center is no longer dependent on one or two key people who carry this information around in their heads. Nor is the data center dependent on the applications staff for day to day scheduling information.

6 With the Calendar File and Scheduling Requirements File, the operations department can "look" at the workload weeks or months ahead. This can be helpful when trying to determine the impact of a new production job or system.

7. Annotated PC and Operater Recap reports provide detailed historical information which can be invaluable in tracking down the source of a problem.

8 Users can easily be provided with copies of current documentation to verify the information the data center uses to process their work.

9 Operations has a clean, simple method for ensuring all documentation has been completed prior to accepting a system "in production". In other words, once the Job Profile Library member has been created and verified, the Data Center Manager knows he has the necessary information to process new systems.

10 New operations personnel can be trained quickly and easily because they no longer need to "learn" how to process a given system or what output to expect from it.

PERFORMANCE CONSIDERATIONS

WYLBUR is the most economical system of its kind because:

- All program code is written in assembler language using efficient coding techniques.
- WYLBUR is a product born of thorough design and careful implementation by a small group of talented technical people.
- Nearly all I/O routines are coded using EXCP, the most efficient access method available. The only exception to this standard is when an installation chooses the VTAM implementation of WYLBUR.
- Only one copy of code exists for all WYLBUR users, not one copy per user as with many systems.

WYLBUR has always had a reputation for low overhead and although OBS has added major features to WYLBUR, we have not neglected performance. Performance is especially critical to good response time and also important when the host environment is in the 370/135 to 148 range. You will find no competitive system with features comparable to WYLBUR which requires less CPU time resources. We are proud that each new release of WYLBUR has resulted in lower resource consumption.

The following chart will give you an idea of what to expect in terms of resource consumption.

CPU	370/158	370/145
Normal Load of Concurrent Users	30	15
Daily CPU Minute Requirement for Entire WYLBUR System	50-70	18-22
Average CPU seconds Per Connect Hour	17-20	15-18

The above consumption statistics are averages based on the actual experience of current installations. The following statistics were taken from one months experience on the OBS 370/158 Mod 1. At the time these statistics were measured, the OBS WYLBUR System supported a peak of 42 concurrent users with an average of about 30 concurrent users. WYLBUR users on the OBS computer tend to be fairly sophisticated WYLBUR users and make extensive use of EXECUTE programs.

Total Connect Hours Provided:	6161
Total Number of Sessions	13,434
Chargeable CPU Time	11.11 CPU Seconds per Connect Hour
Total (Jobstep) CPU Time	16.58 CPU Seconds per Connect Hour
Total Daily CPU Time Used (workdays)	65.5 CPU Minutes
Total CPU Time Used for the Month	1703 CPU Minutes

SECURITY

The principal function of the Access Manager feature is to administer the access via WYLBUR to information belonging to various users. The following paragraphs define in general terms the applicable concepts of ownership and the manner in which access is controlled. It is important to note that the implementation of access control may vary from this general description, since many of these access controls may be selectively implemented by the installation manager.

You may also want to review the sections "IMPLEMENTATION" and "CUSTOMIZING" which occur later in this document. These sections provide more information about WYLBUR system variables and highlight the installation-dependent features such as ownership, naming conventions, user exits, etc.

Ownership

In general, a WYLBUR user is the owner of a file or dataset he creates during a WYLBUR session, or a batch job he submits during a session. In both cases, ownership is reflected in the name which WYLBUR assigns to the dataset or job, since that name will include (in a form selected by the installation) the USERID of the owner.

Installations who desire ownership based on criteria other than USERID may alter this convention. Locally-designed ownership determination code may be placed in the appropriate user exits described in the section "Customizing".

Most installations observe a convention which additionally permits certain datasets to be collectively "owned" by multiple WYLBUR users who constitute a "group". Such collectively-owned datasets may be created by any user within the group. Again, the name of the dataset reflects its ownership, but in this special case, the USERID of a user within the group cannot be utilized to form the name (since that would make the dataset belong to a single user). To deal with this special case, collectively-owned datasets are named as though created by a user within a group who has the special "initials" PUB as part of his USERID

By way of example, assume an installation selects the standard WYLBUR USERID and dataset naming conventions where the five character USERID is composed of a two character group code 'GG' and three-character initials portion 'III'. In this case, the dataset created as a result of the command

```
SAVE MYFILE
```

would be named WYL.GG.III.MYFILE A dataset to be collectively owned by any user within the group 'GG' would be created by either of the following

```
     SAVE &PUB.OURFILE
     SAVE WYL.GG.PUB.OURFILE
```

and the resulting dataset would be named WYL.GG.PUB.OURFILE

Control of Access

In general, the owner of a dataset or batch job may perform any legitimate WYLBUR function which may operate on the dataset or job. For jobs, this includes the capability to FETCH, PRINT, PURGE or CANCEL any particular job For datasets this includes the capability to create, replace, modify or scratch any particular dataset.

Control of Access for Jobs

The following job management functions are subject to access control:

```
          CANCEL
          FETCH
          HOLD
          PRINT
          PURGE
          RELEASE
          ROUTE
```

The owner of a job may perform any of these operations. A user who is not the owner of a job may perform any of these operations upon that job if he can supply the keyword of the owner, or if the user is identified as being authorized in the Access Control Table (ACT).

A user with SYSTEMS Privilege may perform any of these operations on ANY job unless that user is logged on to a dial-up (Switched) terminal.

Control of Access for Datasets

The following functions pertaining to datasets are subject to access control:

```
          CATALOG
          CONDENSE
          COPY FROM
          EXECUTE FROM
          RECATALOG
          SAVE
          SCRATCH
          SHOW DIRECTORY
          SHOW DSNAMES
          SHOW SPACE ON <volume>
          SHOW VOLUMES
          UNCATALOG
          USE
```

Access control to datasets is most basically restricted at the volume level. A user may not access in any way, datasets residing on disk volumes other than those available within his session.

The available volumes are determined by both the "public" volumes specified in the job-step JCL for the WYLBUR system and the interplay of an Access Control Table (ACT) with any "controlled-access" volumes specified in the job-step JCL. The only method by which a user may alter the set of volumes available during his session is to assume the identity of another WYLBUR user via the SET GROUP and SET USER KEYWORD commands (described in Reference Manual). By utilizing these commands, and correctly supplying the keyword for the set USERID, volumes defined for the set userid become available. Since a privileged user may always supply a correct response to the KEYWORD? prompt, he may acquire access to those volumes defined for any other user.

No user, whether privileged or not, may perform any potentially "destructive" operation upon a dataset residing on a disk volume which has the READ ONLY (abbreviated RO) attribute.

Any dataset-related function may potentially be performed for datasets which reside on a disk volume which has the attribute USE AND SAVE (abbreviated U/S). A user who is not the owner of a dataset may perform any of these operations upon a dataset if he can supply the keyword of the owner, or if the user is identified as being authorized in the ACT. Any user may perform any of these operations upon a dataset which is collectively owned by another group if he can identify a user within the owner-group and supply that user's keyword

INDIVIDUAL USERS OR GROUPS OF USERS MAY BE ALLOWED OR DENIED ACCESS TO SPECIFIC DATASETS, GROUPS OF DATASETS (BASED ON INDEX NAME QUALIFIERS) OR ENTIRE VOLUMES BY CODING THE APPROPRIATE MACROS IN THE ACT.

Implementing Access Manager Function

Most Access Manager function is built-in and always in effect, as a result of the WYLBUR system generation. The following are under control of the installation:

- definition of "public" dasd volumes
- definition of "controlled-access" volumes
- creation of an Access Control Table (ACT)

The presence of a usable ACT at WYLBUR system initialization time coupled with the presence of one or more controlled-access volumes activates the Access Manager function for user volume control. If the Access Manager function is not enabled (AM is OFF) then only public volumes are available to any user. If the Access Manager is enabled (AM is ON), then public volumes and designated controlled-access volumes are available to users. In addition, the SET and SHOW AM commands can be used to examine and modify the AM state.

The implementation of the Access Manager provides several additional benefits:

- disk space allowance accounting can be assigned to selected DASD volumes;
- multiple concurrent disk I/O operations can be permitted to the same disk volume (non-VS1 systems only);
- reduced overhead for access to owned datasets;
- ability to utilize special EXECUTE Programs to monitor disk volume access utilization data;
- ability to control on which disk volume WYLBUR recovery will SAVE a user ACTIVE file.

Interface to Security Packages

Installations desiring a WYLBUR interface to packages such as ACF2 and RACF may use the Security Exits described in a later section. Access may be verified through service requests issued in these exit routines.

ACCOUNTING FACILITIES

WYLBUR produces a session accounting record for each terminal session. These records contain statistics concerning resources consumed during the session and generally conform to a TSO logon/logoff record. The installation may choose to have these records written to the system SMF files or to a user defined dataset, or both. Information contained in the record includes session elapsed time, TP and DASD I/O's, CPU time charged and user identification information. A sample DSECT of the Accounting Record follows:

```
*
*         MILTEN ACCOUNTING ROUTINE RECORD
*
*         RECORD WRITTEN AT LOGOFF TIME FOR MILTEN/WYLBUR
*         SESSION. NOTE NO RECORD IS WRITTEN UNLESS
*         LOGOFF OCCURS IN BEHALF OF USER.
*
*         COPY GLOBALS AND CODE SYSDEFN MACRO TO SET GLOBAL SYMBOLS.
*
          SPACE 1
ARREC     DS    0F
          SPACE 1
ARLEN     DS    H                     INCLUSIVE LENGTH
ARRDWBB   DS    H                     X'0000'
ARSIND    DS    XL1                   SYSTEM INDICATOR X'02'
ARTYPE    DS    XL1                   RECORD TYPE CODE (SMF OR OTHER)
ARTIME    DS    BL4                   LOGOFF TIME OF DAY (.01 SECONDS)
ARDATE    DS    PL4                   LOGOFF DATE (00YYDDD+)
ARSER     DS    CL2                   SYSTEM SERIAL FROM CVT
ARMDL     DS    CL2                   SYSTEM MODEL FROM CVT
          SPACE 1
          DS    XL2                    RESERVED
ARTEL     DS    F                     SESSION ELAP TIME (.01 SEC) BIN
ARTPIO    DS    F                     TERMINAL I/O COUNT
ARUID     DS    CL8                   USER ID
ARUG      EQU   ARUID,&IGL               USER GROUP
ARUI      EQU   ARUID+(&IGL),&IIL        USER INITIALS
ARACNO    DS    CL12                  ACCOUNT NUMBER FIELD
ARAN      EQU   ARACNO,&IAL              USER ACCOUNT NUMBER
ARTERMID  DS    CL4                   TERMINAL ID ENTERED AT LOGON
ARCOMPNM  DS    CL8                   MILTEN TERMINAL COMPONENT NAME
ARUCBID   DS    CL3                   UNIT CONTROL BLOCK ID
          DS    XL1                    RESERVED
ARPREL    EQU   *-ARREC               PREFIX LENGTH
          SPACE 1
*         USER WYLBUR USAGE DATA
ARUSER1   DS    0F                    FULLWORD ALLIGNMENT
          DS    F                     RESERVED
```

```
ARWYLTIM DS     F                          CPU EDITING TIME (.01 SEC)
ARWYLPGW DS     H                          NUMBER OF PAGE WRITES
ARWYLPGR DS     H                          NUMBER OF PAGE READS
ARWYLJOB DS     H                          NUMBER OF JOBS SUBMITTED
ARWYLLST DS     H                          NUMBER OF LIST OFFLINES SUBMITTED
ARWYLCND DS     H                          NUMBER OF CONDENSES SUBMITTED
ARWYLOPN DS     H                          NUMBER OF OPENS
ARWYLCAT DS     H                          NUMBER OF CATALOG ACCESSES
ARWYLDAW DS     H                          NUMBER OF DASD NON-PAGE WRITES
ARWYLDAR DS     H                          NUMBER OF DASD NON-PAGE READS
         SPACE 1
ARUSER1L EQU    *-ARUSER1
         DS     CL20                       ...RESERVED FOR EXPANSION
ARLRECL  EQU    *-ARREC                    ACCOUNTING RECORD LENGTH
```

IMPLEMENTATION

Although WYLBUR provides a wide range of powerful capabilities and must interface directly with the Operating System, it is simple to install. No modification to the Operating System is required or desirable for implementation of WYLBUR.

OBS WYLBUR supports parameterized initialization time configuration. The initialization of the WYLBUR system is controlled by several initialization macros, namely the INIT01 and the DEFNSYS macros. Values may be specified for dsname prefix pattern, userid pattern, jobname pattern, number and group of page datasets, number of page buffers, maximum number of concurrent users, job management and so on. These macros effectively allow the installation to design the WYLBUR system according to individual needs and specifications.

The benefits of parameterized initialization include:

- WYLBUR systems which are identical at the load-module level except for those parts with JES or SCP dependencies.
- Distribution of preventive maintenance.
- Users can make adjustments, including those for performance tuning without special knowledge of WYLBUR internals and without the presence of source code.
- Increased responsiveness on the part of OBS in diagnosing any problems encountered in the use of WYLBUR.

JCL for the following jobs is provided with the installation tape and should be used to accomplish the installation of OBS WYLBUR.

1 JOB01 Allocate and catalog the target datasets prior to unloading the primary distribution tape.

2 JOB02 Unload the primary distribution tape into the datasets allocated.

3 JOB03 Link-edit the WYLBUR utility programs into the WYLBUR load library.

4 JOB04 Assemble and link the HASP/JES interface.

5 JOB05 Link-edit all the WYLBUR programs into the WYLBUR Load Library.

6 JOB06 Allocate and initialize the run time datasets

7. JOB07 Create WYLBUR procs.

8. JOB08 Assemble and link-edit the párameterized initialization module.

9. JOB09 Initiate MILTEN/WYLBUR system.

DOCUMENTATION

WYLBUR is well documented and Licensees are provided with complete sets of manuals when the system is delivered. New commands and features are documented in manual updates, when released. Documentation includes:

OBS WYLBUR Reference Summary

A pocket-sized reference to all Commands and options. Special terms are defined and a place is provided for each user to record his or her personalized commands (DEFINEs).

OBS WYLBUR User Guide

An introduction to WYLBUR, this document is readable and provides sample sessions plus other helpful hints on using WYLBUR.

OBS WYLBUR Reference Manual

A detailed description of each WYLBUR Command and parameter. This manual is quite thorough and should be available near each terminal. The first three chapters provide introductory material and a good general description of WYLBUR's 3270 support and Window Mode Full Screen Edit.

OBS WYLBUR Systems and Operations Guide

Intended for use by the WYLBUR coordinator at each installation, this manual contains details on system architecture, system installation, privileged commands, Account and Network definition datasets, the Access Manager, NOTIFY capability, System Utilities, and System messages and codes.

OBS WYLBUR User Exit Guide

The User Exit Guide is a thorough discussion of the various user exit routines supplied by WYLBUR that allow each installation to effectively tailor their system

PAGER User Guide

A thorough discussion and examples of how to use PAGER, the document formatting EXEC provided with your license.

The Reference Manual, User Guide, User Exit Guide and Reference Summary are also distributed in machine readable format on the distribution tape. Once installed, you may print additional copies as necessary. Documentation is also provided by OBS if you wish to order preprinted quantities

OBS WYLBUR USERS GROUP

OBS is supported by a formal Users Group. The OBS WYLBUR Users Group was introduced in early 1979 during the SHARE convention and is open to all OBS WYLBUR installations. This includes installations in the United States, Canada, Europe and the Far East. A broad spectrum of environments are represented in the Users Group, giving the OBS installations an opportunity to trade ideas with installations using an environment similar to their own.

The Users Group evolved after many installations expressed an interest in developing an organized forum to communicate needs and exchange ideas among the user community. The User Group functions independently of OBS, with an elected Steering Committee. An OBS advisor remains on the Steering Committe in a non-voting capacity.

Benefits

• OBS WYLBUR installations have an organized forum for discussion of techniques of training, usage and solutions to common problems.

• Requests for future enhancements to WYLBUR are organized, backed by common needs and prioritized based on general consensus. Because enhancement request are carefully procesed in this manner, the vendor can analyze recommendations of the group for future product directions.

• The information interchange provided through a Users Group has resulted in the development of a library of field developed EXECUTE Programs and documentation that can be shared by all group members.

• Installation profiles have been developed to provide a reference document for member installations to use to contact installations with similar environments for discussion of common problems.

• A user mods tape has been established containing installation developed enhancements to the WYLBUR system. Each installation can obtain a copy of the tape and maximize the benefits of development from other installations as well as enhancements that OBS provides with each WYLBUR release.

• The User Group is represented by an official COLLECT newsletter. The newsletter is periodically issued and contains salient points contributed by individual installations. The philosophy of user/vendor interaction ranks high at OBS, and the newsletter provides another tool in the organized approach our users have.

Summary

In summary, the WYLBUR Users Group provides a vehicle for interaction among users and an open communication line to the vendor This enables installations to be involved in the future directions of their software investment and to benefit from the ideas of other, similar installations.

TRAINING USERS

Ensuring that users know how to effectively use WYLBUR is an important factor in productivity improvement. Formal training is available to WYLBUR Licensees. Normally of two days duration, classes provide an explanation of all important commands and facilities. Terminal sessions are used to allow the student to apply techniques learned in class. Training materials include a notebook for each student. An overhead projector, blackboard, and terminals are required for the course and exercises.

As a supplement to our training program OBS offers a video tape tutorial. The tapes have been graduated into six thirty minute sessions and provide a comprehensive overview of major functions. The video session is accompanied by a training manual and practice terminal sessions.

After training, users will begin using WYLBUR and benefiting from it rapidly. The OBS WYLBUR User Guide is recommended to familiarize new employees with WYLBUR. The Reference Summary is also a handy pocket reference to all of WYLBUR's commands.

Special training sessions, tailored to a specific type of user (Word Processing users or Management Seminar, for example), can usually be arranged to meet any unique requirements.

MAINTENANCE AND SUPPORT

One of the major benefits of software packages is support. No one is more important to OBS than our customers. We strive to make sure that each customer is happy with WYLBUR and using it to capacity. Therefore, if you have a problem with WYLBUR, call us. If you have discovered a bug, we'll fix it. Superzap PTFs will be used in an emergency, followed up by load or source module replacement. When our users call with a problem it is our goal to provide either a fix or temporary bypass before the conversation is over. Fortunately WYLBUR is very stable and not subject to frequent problems.

Maintenance also includes new releases of WYLBUR and Documentation updates. It has been our policy to provide new releases of WYLBUR (even those containing major enhancements such as Window Mode FSE) without additional charge to our customers.

Additionally, the development of parameterized initialization time configuration plays a major role in the stability and responsiveness OBS provides to any problem that might affect you as a user.

CUSTOMIZING

Source code is provided at no additional charge to perpetual licensees. OBS does not encourage source code modifications, however we do encourage installation customizing and have developed a series of user exit routines to support this. An installation can now effectively customize WYLBUR to meet their unique requirements without disturbing the source code.

The user exit routines supplied with WYLBUR are:

1. SONEX - The sign-on exit routine.
2. SOFEX - The sign-off exit routine.
3. KWRREAD - The Keyword Record read routine.
4. KWRWRITE - The Keyword Record write routine.
5. ACCTNOVL - The Account Number validation routine.
6. ACCTWRIT - The Session Accounting Record write routine.
7. WUCMDX - The user command exit routine.
8. WUXDSN01 - The DSNAME construction/modification exit.
9. WUXDSN02 - The DASD Security exit routine.
10. WUXJOB01 - The JOBNAME construction/modification exit.
11. WUXJOB02 - The JES Security exit routine.
12. WUXSUB01 - The Job Submit exit routine.

Each installation is provided with complete documentation when coding the exit routines, and the WYLBUR support team is available to answer questions or provide additional information.

SUMMARY

WYLBUR is the best selection among the many productivity products offered in today's market. The major reasons are:

• Multiplicity of uses - No other system can offer such complete facilities in so many areas:

On-line program development means increased productivity from your programmers.

The Operations Support System means better organization, documentation and ability to respond for your data center.

WYLBUR's Word Processing capabilities offer improved productivity, better accuracy and faster turn-around from your administrative and secretarial personnel.

Document formatting via either PAGER or SCRIPT provides an excellent vehicle to automate your large documents.

Even DP Management can benefit from WYLBUR by using it to store and edit summary material, proposals, white papers, etc.

• Because WYLBUR is so widely accepted by its users, it will help attract and keep the best programming talent. Good workers always enjoy working with the best tools. Users at new WYLBUR installations invariably report satisfaction that management has recognized one of their problems and has implemented an excellent solution.

Because WYLBUR is widely used in Universities, you will find many young programmers already familiar with it.

• Low Operating Cost - OBS WYLBUR offers its features at lower cost in CPU time than any competitive system offering similar capabilities

This means you can provide WYLBUR's services to more users and thus increase productivity over a wider base of people.

Low CPU overhead is also critical to good response time and good response time is extremely critical to user acceptance.

• Stability - No system is worth having if it is subject to frequent outages or maintenance problems. The fact that OBS WYLBUR is offered by many of the largest service companies in the U.S. and Canada attests to its viability and its reputation for stability.

• Price - We're sure you'll agree that WYLBUR is reasonably priced

Low marketing costs and attention to expenses means that your software dollar goes for software, not heavy marketing expenses. Keeping your software budget in line will surely please management as well.

SUPERWYLBUR

OPTIMUM SYSTEMS
INCORPORATED

SUPERWYLBUR

INTRODUCTION (foil 2)

This mornings' talk is organized into the following areas: Interactive Text Editing, which includes the basic command language of Superwylbur; Remote Job Entry and Retrieval, which includes the facilities in Superwylbur for submitting job to the batch system and the subsequent retrieval of the job output for review at the terminal; Document preparation, which concerns the facilities which provide a viable way to prepare documents in Superwylbur; The Macro Facility, which is a complete programming language in Wylbur; and the systems considerations, which concern the devices and operating systems with which Superwylbur can be used.

INTERACTIVE TEXT EDITING (foil 3)

(foil 4)

The basic premise for the creation of Superwylbur is that the text editors available on the market are not very easy to use. SUPERWYLBUR was designed as an easy to use editor whose command language resembles the English language. Our experience in training new users proves that such a goal was valid. Experienced users find that the natural syntax of the command language does not distract from their editing goals.

Input can be in all upper case or in upper and lower case. The user can specify which mode he wishes to use for any portion of a terminal session. With the upper case option, all text is converted internally, and entry can be in capitals or lower case at the terminal. In upper and lower case mode, the case shifts are honored.

The line length allowed is from 0 to 255 characters. This also means we can edit any file that has a logical record size up to 255 bytes. This allows editing of batch files as long as the logical record size is less than 255 and blksize will fit in parameter-specified buffer size.

We allow for several thousand lines in a working file. The working file refers to the copy of any data set internally in WYLBUR. The capacity of a working file actually depends on the density of the material. If you have a lot of blanks, there will be more lines than if there is a great deal of material on each line. An estimate of the capacity is around 10,000 lines in each working file.

We also allow multiple working files called temporaries (abbreviated t or temp). Any command to which a t or temp and the name of the temporary is appended operates on that temporary data set. It is thus possible to have, for example, 3 or 4 programs or documents, and all the commands can work on any of these.

FACILITIES FOR ADDRESSING TEXT (foil 5)

Line number directed. (foil 6)

It is necessary at this point to discuss the facilities which exist in SUPERWYLBUR for specifying what data the editor is to operate on. If you would consider that the following items are specified as the operand of a list command (entered as list or l), it may help in your understanding.

The first way to specify the data on which to operate is by the line number associated with the line in the working file. It should be noted that line numbers are assigned when data enters the system and renumbering is done only when requested by the user. There is an eight digit line number, that is five digits followed by three decimal places.

As in the first example, the line number can be single, in a range, or first or last.

There is also a current line pointer. Current (abbreviated c) is an addressable line. For example, if you are entering text and you see a line that you have entered which is incorrect, you can get back in command mode and say modify c which stands for modify the current line. The modify command will be explained later in the talk. The current line is updated by collect, modify, copy, move, insert, delete or point, all of which will be discussed later in this presentation. We can set up a range of lines, i.e. a segment of the program, as in the first of the two examples as being bounded by the current line through the line numbered 400.

Relative addressing is also allowed.

Relative addressing refers to the syntax illustrated in which we specify an existing starting point, plus or minus a relative line position. For example, the first +3 lines will be the fourth line in any data set regardless of what the line numbers of that data set are. It is not the first line plus three in any kind of line number sense. It is strictly relational. In the second example, (100-7/150+5), the lines that we would operate on will be seven lines before line 100, whatever that line number is through five lines after line 150, whatever line number that is. I might mention that you can make any combination of these terms in a single command.

Content directed (foil 7)

The ability also exists to look at, or select, things by content of the line. The first example is to look for any line which has that particular string 'HAPPY' or narrowing the possible candidates as in the 2nd example by specifying it is to begin in a column 7 or in the third example by specifying the string is contained between columns 10 and 20. And finally, by a combination of criteria directing the search to either a string beginning in column 7 or contained between columns 15 and 30.

(foil 8)

Content can also be defined in terms of classes of things which you can list. For example: List any digit in column 10. Also, it is possible to concatenate operands as in list 'AB' concatenated with a digit. There is no concatentation symbol in the syntax, so that concatenated items are just placed one after the other. There is also a class which is a blank. The problem there is if you want to change the string 'the' in a document using many editors, you can't say change 'the' to 'XXX' because not only the "the's", but the "there's", etc. would be changed. So you are forced to say change ' the ' to 'XXX'. That changed most of what you wanted, unless 'the' happened to occur at the beginning or the end of the line. What you can now put in is list blank concatenated with 'the' concatenated with blank and that means the string that I set up now can find either a blank or the beginning or end of the line with the string 'THE'.

(foil 9)

The pattern-oriented operators are ways of defining or narrowing the search for a content operator. Let's look at the examples;

```
DIGIT NOT (8 OR 9)
```

That would be an octal digit.

Or in the second

```
(DIGIT*2 or UPPER*2) DIGIT ' ' DIGIT*4
```

example which would represent a telephone number minus the area code.

Set operators use union, intersection, complement. Set operators operate on multiple lines. An example:

```
'OPEN' 10 INT 'INPUT'
```

With 'intersection' you can put multiple constraints on a string by saying list 'A' in 'B' in 'C' in 'D'. As long as I keep using 'in', I can continue to narrow the search. 'In' and 'intersection' frequently give the same result.

(foil 10)

We can also divide things by instance of occurrance. An example:

```
3RD AFTER 'ABC' 1
```

the third line after each line containing 'ABC' in column 1. The second example:

```
EVERY 2ND AFTER 50TH of '$'
```

every second line (every other) line after the fifteth line containing a $.

(foil 11)

As I mentioned earlier, these terms can be combined. Please observe the following examples.

The first example has lines between the number 200 and 500 in which the word 'balance' is contained between columns 10 and 30.

In our second example, those lines where ABC starts in column 7 or 40 or 72 in the 4th line through the line numbered 25.

The result in the third and fourth examples, in both cases, is those lines which have RALPH beginning in column 10 and also have GEORGE beginning in column 28 lying between line number 200 and the next to last line in the file.

The last example, all one command, lists the first line after the line that has 'TH' in column 10 and 'FLAGA' in column 10 and 'FLAGAON' somewhere else on the line in every line following that line where 'symbol' occurs in column 1.

SUPERWYLBUR COMMANDS. (foil 12)

We will now look at the command language of SUPERWYLBUR. Time does not allow a great amount of detail on each, so I will attempt to hit the high points.

(foil 12.5)

A single command can address more than one working file. I can say change 'A' to 'B' in Temp A and Temp B and Temp C and Temp D and Temp E, etc. The change command would make the change in all of them. We also allow commands to operate on permanent storage - you can say RUN FROM DATASET. You don't say use and run.

The Line Commands (foil 13)

The first command is collect. Collect (abbreviated col or c) is the request to enter data starting at a specified line number, optionally specifying the increment by which the line number should increase.

The system will prompt with the line numbers or, with the unnumbered option, merely unlock the keyboard for entry.

If directed to start collecting input at a line which already exists, Superwylbur will add an increment to the number you specified and begin collecting just after the specified starting point.

An option of collect which is not on the foils is The command option of collect. If you enter collect temp A command show dsns on vol, what would have printed on the terminal, from the command show dsns on vol, is put into working file A. Any command can be done on the collect command except another collect command or one which destroys the temporary the output is directed to.

Insert, delete, and replace do what you would expect.

The short form alluded to at the bottom of the foil refers to a convention that a line number followed by text (entered as a command) is an addition or replacement of that line in the default working file. A line number without text is taken to be a request to delete that line in the default working file.

Intra-line Editing (foil 14)

The change command takes any string or pattern definition (as described in the first few foils) or column replacement of the form CH 5/7 TO or column replacement of the form CH 5 TO. The replacement can be a quoted string or the word HEX followed by hexidecimal digits or an incremented integer (of the form CH 75/80 to 10 + 10) or the reserved words UPPER (meaning translate to upper case) or LOWER (meaning translate to lower case).

We also can use a transferred replacement where the replacement data can exist as lines of the file.

We also can specify more than one change to be made at the same time with the AND option.

Note that in the first example replacement strings need not be the same length as target strings with the system shifting text in the line as directed.

The first command changes every occurance of the string 'AB' to the string 'DEF'.

The second example would change any 2 digit number contained in columns 15/17 to the hexidecimal string 404BFF.

The third example would change the string XYZ to the null string from any line where it started in column 10 in the range from line number 400 through the last line of the file. This would remove the 'XYZ''s and shift the lines accordingly.

The fourth would insert an 'X' at column 10 in those lines containing the string 'ABLE' and having two zeroes in column one.

The fifth would delete column position 14/16 and insert in their place a slash.

The next to last is an example of a transferred replacement where the replacement data is the lines containing the string 'XYZ' in line 500 through last and the target is those lines containing the string 'ABC' in the line zero through line 499.

And the last where AB is replaced by DEF and 123 is replaced by 89 in lines which do not begin with an asterisk.

The modify command is a unique method of making changes to a line or group of lines in which the system prompts with the text of the line and changes are indicated by moving the type element to the position to be acted on. Then by entering a code letter like d for delete or i for insert or r for replace, to name just a couple, followed by the data to be inserted or the replacement text. The system will then update the line and reprompt until the user indicates that all changes to that line are finished.

(foil 15)

The COPY and MOVE commands can operate on lines in one working file or between multiple working files or between disk data-sets and one or more working files.

The 'ditto' option specifies how many times the data that is MOVE'ed or COPY'ed is to be repeated at the destination.

The 'AND' option allows specification of multiple destination (i.e. copy 1/5 to 5 TEMP A AND 10 TEMP B).

The 'SHORT' option requests the system to list only the first message from the 'COPY' or 'MOVE' command.

The 'AFTER' and 'BEFORE' options can be used to specify that the data is to be MOVE'ed or COPY'ed after or before each line in a second range.

The 'COPY' command normally does not allow interleaving or replacement of lines based on line numbers. The 'MERGE' option will allow interleaving but not replacement. The 'OVERLAY' option allows both interleaving and replacement. The first example then would COPY lines numbered one through 10 to line 50.

The second example would COPY all lines in the data set SOURCE.A to the end of the default working file. The third example would MOVE all lines containing the string 74105 to the end.

The last example would COPY lines numbered 1 through 30 to line 15 and line 150.

The MERGE and OVERLAY commands operate in a manner similar to COPY with MERGE and OVERLAY options as previously explained.

The list command causes the data requested to be listed at the terminal or optionally on the printer.

The POINT command operates the same as the LIST command but also resets the current line value.

The RETRY command causes the last command entered to be prompted in the same manner as the modify command.

The COMPARE command matches two ranges, without regard for line numbers, and notes dissimilarities.

The remainder of items are CLEAR, SET, and SHOW commands for various fairly straightforward options.

The data set commands. (foil 17)

(foil 18)

The USE command causes a disk set to be loaded into a working file. Any commands operate on the copy of the data set in working storage. All characteristics of the data set are retained.

The SAVE command unloads the contents of a working file to a disk data set. All original characteristics which are not overridden are used.

The RESAVE command causes the working file to be unloaded and any existing data set with the dsname to be scratched.

Typically, if you are making changes in a program you say use temp x from data set, you put in your changes and say resave.

Resave on catalog and we go back and put it on the pack and update the catalog. If I say scratch some data set on catalog it will scratch the data set and take it out of catalog. If you say rename x on catalog, the processing includes all the functions required to do that task correctly.

The 'RENAME', 'CATALOG', 'RECATALOG' and 'UNCATLG' command perform the expected operation on the O/S catalog.

The SHOW DSNAMES and SHOW MEMBERS commands list the data sets or members and optionally give a full readout similar to the LISTVTOC FORMAT or LISTFDS output of IEHLIST.

The SET PREFIX command allows an automatic index level or levels to precede the data set names used in any command.

The SET VOLUME command establishes a default volume for a user and data set commands in which no volume is given are assumed to operate with the default volume. The SHOW VOLUME command shows what the default volume is, if any.

There is also show space. Enter show space on volser. Superwylbur will display the current status of free space on that volume.

(foil 19)

In summary, SUPERWYLBUR can access both sequential and partitioned O/S data sets. All data that is SAVE'd, and no lrecl is specified, is kept with all blanks removed. This typically saves about 60% space for source language data sets.

Optionally the data can be SAVE'd in a coded format.

The encoding algorithm is the algorithm used by the State Department for classified material.

As I stated earlier, the LIST command can optionally be directed to the printer. Our HASP interface allows ROUTE'ing and does not involve the execution of any batch program.

In the event of either the phone line dropping or the system going down, SUPERWYLBUR will recover the data up to the last command entered.

Automatic recovery of system failures with Superwylbur not only goes out and gets your files but also gets all your temporary files, plus user commands and options such as tabs, length and volume, prefix, etc.

Automatic setting of user options.

Superwylbur will look for a predefined PDS member or seq. file and automatically calls that file if there wasn't any recovery file.

When you logon either you get recovered to where you were when you hung up or your PROFILE dataset is called automatically.

REMOTE JOB ENTRY FUNCTIONS. (foil 21)

(foil 22)

The RUN command causes the working file specified to be passed to HASP for execution.

When a job has completed, the output (either all or a specific ddname) can be retrieved by use of the FETCH command. The output is placed in a working file, and any editing commands can be used to inspect the results. If the output is not as expected (perhaps compiler errors), it can be PURGE'd and hence never get to the printer. This facility results in a dramatic reduction in printer load and paper savings (perhaps up to several thousand lines). This is even more dramatic where slow RJE printers are involved.

The jobs status can be obtained and changed (internally using standard HASP commands) after ownership of the job is checked.

Output can be ROUTE'd to a HASP remote station. Also, optionally the SET REMOTE command can default all output to a specific remote.

I might also mention that our Production Control personnel make heavy use of SUPERWYLBUR. The usefulness of the system for helping in the production control problem is well proven.

DOCUMENT PREPARATION (foil 24)

(foil 25)

Getting into document preparation capabilities,- We are compatible with IBM's ATS and many of the commercially available ATS derivatives.

The system allows alignment or justification of the right margin.

Lines can be centered or positioned at the right or left margin.

SUPERWYLBUR supports pagination with complete page headings and page footing, including separate even and odd headings and footings.

It should be noted that all commands handle underlining or overstriking properly.

The system also includes an algorithm which optionally HYPHENATES text in the alignment or justification process. Our most common use of this facility is in a mode in which all hyphenations are prompted before being included. The prompt includes an indication of where the algorithm has indicated the hyphen should be plus where the ideal end of line would be. The terminal operator can then accept or move the hyphen; or review what the line looked like originally and what the newly formatted line looks like; or that no hyphenation of that word is desired. Our experience to date is that 80 - 85% of the algorithmically derived hyphens are correct.

The facility exists to stop the listing on the terminal to allow changes in type elements or paper or to insert additional text.

All listing on the printer can request special forms or trains. All underlining and overstriking is handled correctly.

SUPERWYLBUR also allows for automatic generation of table of contents type data. The technique may also be helpful for limited index generation.

THE MACRO FACILITY (foil 30)

The macro facility is really a complete programming language which in conjunction with the command language we have previously discussed, constitutes an extremely powerful and useful addition to the SUPERWYLBUR system. If you are not familiar with a programming language written in a text editor, I remind you of the CLIST language of TSO, the EXEC file language of CMS, or the LOAD FILE language of some WYLBUR'S which are all primitive implementations of what was truly needed. SUPERWYLBUR offers far greater power and flexibility than any IBM machine editor to date.

(foil 51)

The facility lends itself to problems such as interactive data entry with validity checking and/or reformatting, or the problem of constructing parameter cards for non-programmer users. The macro facility includes the 'TYPE' command where the command is 'TYPE' (or t) followed by any mathematical expression. Superwylbur handles floating point and exponential notation and does arithmetic to 15 significant digits, thereby providing a desk calculator capability.

There also is the SET COMMAND facility which gives you the ability to add your own commands or synonyms. You would enter Set command A : some command(s). Then when you type 'A' the command(s) will be executed. Any commands that you set are recovered, by the same recovery mechanisms discussed earlier.

The macro language has variable names formed by any 16 char combination of alphanumerics and periods. You can get nice readable variable names. They are not typed variables and can contain either strings or numbers.

Data can be entered as integers or as decimals or as exponents. Any number that is going to print larger than 10 digits is automatically converted to exponential form.

Conditional branching is complete. The macro syntax includes an if, else, structure. There is a match operation.

There is the full ability to read or write from the terminal and that's extended so that you can specify your own prompting.

There is the full ability to read or write from working files. They can be addressed with an INPUT and an OUTPUT statement or an INPUT and an UPDATE statement or any of the commands of the text editor.

You can preceed any command with the QUIET option. That stops all but problem error messages from being output on the terminal. If you are developing a macro that you are going to have someone use who doesn't understand computers, you can mask the responses.

Many functions are provided, but time does not allow a thorough explanation of them. Suffice it to say that functions are included providing a complete summary of data having to do with the terminal session, such as the date, time, day of the week, etc. Also many string and numeric functions, for example SUBSTRING, INDEX, absolute value, etc. In addition, there are conversion functions for hex to string, string to number, etc. And lastly, those functions needed for access to the working files, such as .LINE(i) (returns the text of line i) or .CURRENT(w) (current line number in w).

SYSTEM CONSIDERATIONS. (foil 30)

(foil 38)

Superwylbur runs under O/S MVT or MFT with HASP 3.1.

The system will run under TSO (as a command) or outside TSO. All code in the system is fully re-entrant, and placement in the TSO linkpack or the O/S linkpack would allow concurrent use as a command and separate task. We currently can run under VS2 release 1, but the HASP4 interface is not yet complete.

Support for ASP and VS1 (as a CRJE plug-compatable replacement) are, underway and VSAM and VTAM will be tackled when we're sure they really work.

(foil 39)

The system is written to support 1 to 250 simultaneous users and support is practical for 150 to 200 without fully saturating a 165/8 CPU while providing response time mostly under one,second.

The cycles taken are highly dependent on number of terminals and work being done, but CPU usage of about 8-10% of a 65 would support about 30 terminals.

The system will handle about 30 terminals at 170K plus the terminal interface (MILTEN) at about 70K.

An overlay structure based on the HASP overlay technique allows paged code for low activity functions, and 97 overlays are currently used. The overlay usage is measured, and tuning can maintain good response with a large amount of paged code.

(foil 40)

When run under TSO, the terminals are those supported by TCAM. With MILTEN, which is an EXCP level teleprocessing system in use at some 50 installations for over five years, support exists for EBCDIC or correspondence 2741's, teletype model 33, 35 and 37, and those countless teletype compatables.

(foil 41)

As you've seen we have full support for the O/S catalog.

Any direct access device can be used for paging or permanent storage.

RPS is used where available.

Multiple SUPERWYLBUR page files can exist on same or different devices.

The system includes 370 instructions only when generated for a 370.

SUPERWYLBUR

- INTERACTIVE TEXT EDITING
- REMOTE JOB ENTRY & RETREIVAL
- DOCUMENT PREPARATION
- MACRO FACILITY
- SYSTEMS CONSIDERATIONS

INTERACTIVE TEXT EDITING

- ENGLISH LANGUAGE-LIKE COMMAND STRUCTURE
- UPPER CASE ONLY OR UPPER AND LOWER CASE INPUT OPTION FOR EACH USER
- 0 TO 255 CHARACTERS PER LINE
- SEVERAL THOUSAND LINES PER WORKING FILE
- MULTIPLE WORKING FILES FOR EACH USER

FACILITIES FOR ADDRESSING TEXT

- LINE NUMBERS (8 DIGITS : DDDDD.DDD)

  I.E.
  ```
  10
  150/250
  10, 150/250, 400, 600/LAST
  FIRST/50, 1000
  ```

- CURRENT LINE POINTER

  I.E.
  ```
  CURRENT/400
  ```

- RELATIVE ADDRESSING

  I.E.
  ```
  FIRST+3
  100-7/150+5
  ```

- CONTENT

CHARACTER STRINGS

I.E.	'HAPPY'		
	'DO'	7	STARTING COLUMN
	'SAVE'	10/20	CONTAINING COLUMNS
	'DATA'	7,15/30	BOTH

- CONTENT

CHARACTER CLASSES

ANY	- ANY CHARACTER INCLUDING BLANK
DIGIT	- ANY DIGIT (0 THRU 9)
LETTER	- ANY ALPHABETIC CHARACTER (UPPER OR LOWER CASE)
UPPER	- ANY UPPER CASE ALPHABETIC CHARACTER
UPLOW 'string'	- ANY STRING WITH 'string' CHARACTERS EITHER UPPER OR LOWER CASE ('AT' FOR At, AT, at, aT)
PUNCTUATION	- PUNCTUATION CHARACTERS
BOL	- BEGINNING OF LINE
EOL	- END OF LINE
BLANK	- BLANK CHARACTER ' ' OR BOL OR EOL
HEX 'hex-string'	- CHARACTER STRING EQUIVALENT TO THE 'hex-string'
EMPTY	- BLANK LINE
BELOW 'string'	- STRING BELOW 'string' IN COLLATING SEQUENCE
ABOVE 'string'	- STRING ABOVE 'string' IN COLLATING SEQUENCE
SOME 'string'	- OCCURRENCE OF ANY ITEM IN 'string'

I. E. DIGIT 10

- CONTENT

  BOOLEAN COMBINATIONS OF STRINGS AND CHARACTER CLASSES

  PATTERN ORIENTED OPERATORS

  AND

  OR

  NOT

  DUPLICATION FACTOR

  CONCATENATION

  THRU

```
I.E.  DIGIT AND NOT ('8' OR '9')
     (DIGIT*2 OR UPPER*2) DIGIT '-'
      DIGIT*4
```

  SET OPERATORS

  UNION

  INTERSECTION

  COMPLEMENT

  SET

```
I.E.    'OPEN' 16 INT 'INPUT'
```

- INSTANCES OF LINES

  ORDINALS 1ST, 2ND, 3RD, NTH

  OF

  AFTER

  BEFORE

  EVERY

  BETWEEN

  START

```
I.E.    3RD AFTER 'ABC' 1
        EVERY 2ND AFTER 50TH OF '$'
```

- COMBINATIONS

```
'BALANCE' 10/30 IN 200/500

'ABC'  7, 40, 72 IN FIRST+3/25

'RALPH'  10 IN 'GEORGE' 28 IN 200/LAST-1

'RALPH'  10 INT 'GEORGE' 28 IN 200/LAST-1

1ST AFTER 'TM' 10 IN 'FLAGA' 16
IN 'FLAGAON'21/LAST IN EVERY 1ST AFTER 'SYMBOL' 1
```

SUPERWYLBUR COMMANDS

- SINGLE COMMAND MAY ADDRESS MORE THAN ONE WORKING FILE

- COMMANDS MAY ADDRESS FILES ON PERMANENT STORAGE AS WELL AS WORKING FILES

- LINE COMMANDS

```
COLLECT <lineno> <BY   increment>

INSERT   lineno  <lineno2...linenon> <DITTO>

DELETE   range  <LIST>

REPLACE   range  <DITTO>  <LIST>
```

SHORT FORM FOR INSERT, DELETE, REPLACE

- INTRA-LINE EDITING

  - CHANGE target TO replacement

    WHERE target: STRING OR PATTERN
    COLUMN REPLACEMENT
    COLUMN INSERTION

    replacement: QUOTED STRING
    HEX string
    INCREMENTED INTEGER
    UPPER and LOWER

    TRANSFERRED REPLACEMENT
    AND

```
i.e.    CHANGE 'AB' TO 'DEF'
        CH DIGIT*2  13/17  to  HEX 404BFF
        CH 'XYZ' 10  TO  ''  IN  400/LAST
        CH 10  TO  'X'  IN  'ABLE'  IN  '00'  1
        CH 14/16  TO  '/'
        CH 'ABC'  USING  'XYZ'  IN  500/LAST  IN 0/499
        CH 'AB'  TO  'DEF'  AND  '123'  TO '89'  IN
            NOT  '*'  1
```

  - MODIFY

- COPY and MOVE

  DITTO
  AND
  SHORT
  AFTER
  BEFORE
  MERGE
  OVERLAY

```
I.E.  COPY 1/10 TO 50
      COPY ALL FROM SOURCE.A TO END
      MOVE '74103' TO END
      COPY 1/30 TO 15, 150
```

- MERGE

- OVERLAY

- LIST
- POINT
- NUMBER
- RETRY
- COMPARE
- CLEAR

◦ SET UPPER	SHOW CASE
UPLOW	
◦ SET LENGTH	SHOW LENGTH
◦ SET TABS	SHOW TABS

- SHOW COLUMNS
- SHOW SIZE
- SHOW STATE

DATA SET MANAGEMENT COMMANDS

- USE
- SAVE
- RESAVE
- SCRATCH
- RENAME
- CATALOG
- RECATALOG
- UNCATALOG
- SHOW DSNAMES
- SHOW MEMBERS
- SET PREFIX SHOW PREFIX
- SET VOLUME SHOW VOLUME
- SHOW SPACE

- ACCESS TO BOTH SEQUENTIAL AND PARTITIONED O/S DATA SETS
- COMPRESSED DATA SET STORAGE
- ENCODED STORAGE FORMAT FOR SENSITIVE DATA
- HIGH-SPEED PRINTER LISTINGS
- AUTOMATIC RECOVERY OF SYSTEM FAILURES
- AUTOMATIC SETTING OF USER OPTIONS
- ABILITY TO MODIFY PREVIOUS COMMAND

REMOTE JOB ENTRY & RETRIEVAL

- JOB SUBMISSION
- OPTIONAL RETRIEVAL OF OUTPUT AT TERMINAL
- ALL EDITING COMMANDS AVAILABLE TO SCAN OUTPUT
- JOB STATUS INQUIRY
- FACILITY FOR CHANGING JOB STATUS
- OUTPUT MAY BE DIRECTED TO A REMOTE STATION

DOCUMENT PREPARATION

- ALIGNMENT OR JUSTIFICATION OF RIGHT MARGIN
- CENTERING AND POSITIONING OF LINES
- AUTOMATIC PAGINATION WITH:

  A) PAGE NUMBERING

  B) PAGE HEADINGS

  C) PAGE FOOTINGS
- FULL SUPPORT FOR UNDERLINING AND OVERSTRIKING
- OPTIONAL HYPHENATION
- SUPPORT FOR SUSPENDING OUTPUT IN ORDER TO:

  A) CHANGE TYPE ELEMENTS

  B) CHANGE PAPER

  C) INSERT TEXT
- ABILITY TO LIST ON HIGH-SPEED PRINTER (WITH SPECIAL FORMS AND TRAINS) ALL UNDERLINING AND OVERSTRIKING INCLUDED
- AUTOMATIC TABLE-OF-CONTENTS GENERATION

MACRO FACILITY

- USEFUL FOR DATA-ENTRY VALIDITY CHECKING AND REFORMATTING
- USEFUL FOR AUTOMATIC CONSTRUCTION OF JCL AND PARAMETER CARDS FOR NON-PROGRAMMERS
- USEFUL AS A DESK CALCULATOR
- ABILITY TO ADD USER-DEFINED COMMANDS AND SYNONYMS
- ABILITY TO EXECUTE PRE-STORED COMMANDS
- NUMBER AND STRING VARIABLES
- CONDITIONAL BRANCHING
- ABILITY TO READ OR WRITE TERMINAL
- ABILITY TO READ OR WRITE WORKING FILES
- NESTING OF MACRO CALLS

SYSTEMS CONSIDERATIONS

- RUNS UNDER O/S MVT, MFT WITH HASP 3.1
- RUNS UNDER TSO, OUTSIDE TSO, OR BOTH
- PLANNED SUPPORT FOR ASP, VS2, VS1, VSAM, VTAM

- SUPPORT 1-250 SIMULTANEOUS USERS

- RESPONSE TIME MOSTLY UNDER ONE SECOND

- LITTLE OR NO EFFECT ON BATCH THROUGHPUT

- MINIMUM CORE REQUIREMENTS APPROXIMATELY 170K

- PAGED CODE FOR LOW-ACTIVITY FUNCTIONS

- SUPPORT FOR:

  A) EBCDIC OR CORRESPONDENCE 2741's

  B) TELETYPE MODEL 33, 35, 37

  C) TELETYPE COMPATIBLES

- SUPPORTS O/S CATALOG
- SUPPORTS ANY DIRECT ACCESS DEVICE FOR PAGING AND PERMANENT STORAGE
- SUPPORTS ROTATIONAL POSITION SENSING
- SUPPORTS MULTIPLE PAGING FILES

ROSCOE

ROSCOE
Applied Data Research, Inc.

MANAGEMENT SUMMARY

ROSCOE (Remote OS Conversational Operating Environment) is a low-overhead, comprehensive, on-line program development system that contains a wide variety of interactive special functions not found in IBM's TSO (Time Sharing Option). ROSCOE can be installed on any IBM, or compatible, computer capable of running the OS, VS1, SVS, or MVS operating system. As previously mentioned, ROSCOE competes favorably against TSO. TSO is a generalized time-sharing system as compared to the multi-functioned ROSCOE. ROSCOE also vies favorably against IBM's CMS (Conversational Monitor System) under VM (Virtual Machine) /370, requiring significantly fewer resources to perform comparable tasks.

ROSCOE incorporates a wide range of facilities to serve the needs of application and system programmers as well as computer users who are not programmers. Some of these include: full on-line data entry and editing facilities; an efficient library system for permanent data storage; full facilities for job entry and job output retrieval in any OS/VS system; a hierarchical system of security features; interactive processing for performing frequently used functions; syntax checkers for COBOL, FORTRAN, PL/I and Job Control Language (JCL), and a monitor with loading, execution, and I/O services for user-written programs.

In addition to the many standard features found in ROSCOE, some additional cost options are available. COBOL/ADE is a structured system option for COBOL debugging, and MINIBASIC is a special implementation of the BASIC problem-solving language. RTS is an option that can be implemented for problem solving in time-sharing mode. The latest version of ROSCOE contains improved security measures for data protection, as well as added reporting facilities to keep an accurate account of system and human resource utilization.

USER REACTION

In the 1978 Datapro survey of proprietary software users, 20 respondents rated ROSCOE. As a result of these ratings, ROSCOE earned a place on the Datapro Software Honor Roll for the second consecutive year. The ratings are shown in the table below.

	Excellent	Good	Fair	Poor	WA*
Overall satisfaction	11	8	1	0	3.5
Throughput/efficiency	10	10	0	0	3.5
Ease of installation	9	8	3	0	3.3
Ease of use	12	6	2	0	3.5
Documentation	5	11	4	0	3.1
Vendor technical support	4	11	5	0	3.0
Training	3	13	3	0	3.0

*Weighted Average on a scale of 4.0 for Excellent.

Nineteen of the 20 respondents identified their computers: there were nine System/370 Model 158's, two Model 138's, two Model 148's, two Model 155's, one 3032 processor, one System/360 Model 50, one Model 65, and one Itel AS 5 system. Nine users were running under VS1, five under OS/MVT, three were using MVS, and two had the SVS operating system. The average ROSCOE usage ▷

A remote job entry system that provides a conversational approach to program preparation, testing and maintenance via remote terminal facilities. It is usable on any IBM System/360, 370, 303X, 4300, or compatible computer running under OS or OS/VS.

A permanent license for basic ROSCOE costs $29,000, while a permanent license for ROSCOE/MVS costs $33,000. Monthly licenses for these are $1,610 and $1,840 respectively.

Currently there are approximately 650 users.

CHARACTERISTICS

VENDOR: Applied Data Research, Inc., Route 206 Center, CN-8, Princeton, New Jersey 08540. Telephone (609) 924-9100.

INITIAL INSTALLATION: June 1970; current version was released in February 1979.

CURRENT USERS: Approximately 650 as of June 1979.

BASIC FUNCTION: ROSCOE is a remote job entry system that provides remote terminal users with full access to an IBM computer under OS or OS/VS for program development, testing, and maintenance. It possesses on-line data entry and editing facilities, sophisticated security capabilities, and complete data management facilities for application and system programmers. ROSCOE provides syntax checkers for COBOL, FORTRAN, PL/1, and the JCL. There are a structure for COBOL debugging and a special implementation of the BASIC language processor available as options.

OPERATION: ROSCOE is loaded and terminated through the computer operator console as a "never-ending" job in a partition or region under IBM's OS. Numerous programs can be developed concurrently from multiple remote terminals under ROSCOE, with an Active Work Space (AWS) associated with each terminal. A conversational command system allows the user to manipulate his AWS area and library files, with job submission, initiation, and output recovery all controlled at the remote location.

As each user is logged into the system, AWS and library storage areas are assigned. ROSCOE drops all spaces and blanks, thus reducing library storage space requirements.

The AWS handles card-image data. Each AWS line has an internal line number associated with it. Manipulation or editing of the AWS is done either implicitly by line number or through explicit conversational ROSCOE commands. These commands permit assignment or renumbering of line numbers; display, modification, insertion, or deletion of all or part of the AWS by line number; character string manipulation for single or multiple lines; and tab settings so that remote keyboard data entry is automatically placed at user-specified positions in the designated AWS line.

The User Library is organized into named data sets and provides permanent storage facilities for user data. Individual data sets are protected by a prefix associated with the user's ROSCOE authorization code. Each user data set is manipulated as a unit, and there are library commands which allow the transfer of complete data sets from the AWS to the library and vice versa. ➤

time was 16.9 months, with nine users reporting having had ROSCOE for one year or less at the time of the survey.

The advantages checked off by these respondents showed that 16 realized savings in human resources, 11 reported savings in system resources, 10 judged it flexible, and 2 said it is inexpensive. The number of users checking off disadvantages was small in comparison. Two felt it is costly, one each stated that it is complex, uses excessive resources, and lacks key capabilities.

None of the ROSCOE users included any comments on their returned questionnaire form, obviously feeling that their ratings clearly express their satisfaction with the product. Looking at these ratings, one notices that all the final weighted average ratings are 3.0 or better; a solid commendation for ROSCOE. The ratings on documentation, vendor technical support and training show that some improvement could be realized in these categories.

With testimonials in the form of the submitted ratings, which returned ROSCOE to the Datapro Software Honor Roll for the second consecutive year, we feel that any OS or OS/VS installation looking for an effective, resource-saving conversational monitoring system would benefit from evaluating what ROSCOE has to offer.□

The data sets belonging to each ROSCOE user are protected by a hierarchy of mechanisms: the user's key and password (which are maintained on a scrambled file), internal protection code, and protection groups (optional). These prevent unauthorized access or modification of each user's data sets. The ROSCOE facilities themselves are secured by a system of user exits which permit management to define and protect sensitive areas on a site basis.

Frequently used sequences of ROSCOE commands can be stored in the User Library under a data set name. Such a sequence of commands is known as a ROSCOE conversational procedure, or ROSPROC, and can contain a certain amount of logic or decision-making capability. Of particular interest is the CALL command that permits one ROSCOE procedure to call another, with linkage beginning and ending at user-specified statements in the called ROSPROC.

Actual control over the running of users' jobs is handled through the SUBMIT command and the Output routine. The SUBMIT command enters a single job or a jobstream to OS for batch processing, with appropriate JCL either entered through the remote terminal keyboard or fetched from a user library data set under remote terminal control.

The Output routine permits the ROSCOE user to view an entire job's output directly from the system queues. It contains commands to facilitate rapid browsing, with options to place selected portions of the output in the AWS and to purge an entire job, or portions thereof, after viewing.

The ROSCOE User Services are a set of major subsystems extending the basic ROSCOE facilities. All of these services run under control of the ROSCOE Monitor, a standard facility with services for loading, execution, scheduling, and ROSCOE I/O operations. The Monitor can accommodate up to 255 routines. Among these services are the syntax checkers, LIB/OL for on-line execution of ADR's LIBRARIAN, AMS for VSAM data management functions, the COPY routine for fetching OS data sets into the AWS, SUPERZAP for modifying and verifying direct-access data sets, ADR's LOOK system, and a series of applications-oriented facilities.

For use by management, systems programmers, and even end users, ROSCOE produces several different kinds of accounting reports. These reports can describe terminal activity by user, project, and date; ROSCOE resource and I/O usage; library usage; and the ROSCOE system configuration. There is a set of user exits that permit site-written routines to supplement or replace the standard ROSCOE reports.

RTS (ROSCOE Interactive Time-Sharing Monitor) is an extra-cost option that works in a VS environment to permit execution in time-sharing mode of programs placed in a special time-sharing library. The programs can be written in FORTRAN, PL/I or BAL. There is no limit on the number of programs in the library nor on the number in concurrent execution at any time, except for that imposed by storage limitations. A program to be executed under RTS is first debugged and tested using standard ROSCOE facilities. When a clean working version is obtained, it is put in the time-sharing library. This last step is performed by the ROSCOE system manager, who creates a table entry for each such program describing its name, language, and file requirements. The program, without modification to the original code, is link-edited with a time-sharing front end and placed in the library, where it is then available for on-line ROSCOE execution. ROSCOE then passes control to RTS. Using its own command language, RTS requests from the terminal user the name of the routine to be executed and places it in execution by attaching it as a subtask.

Two extra-cost features incorporated into a recent release of ROSCOE are COBOL/ADE and MINIBASIC. COBOL/ADE (Automated Debugging Environment) is a facility for dynamic COBOL program verification that provides the tools and a carefully controlled environment for easier program testing and debugging. COBOL/ADE simulates those system programmer activities that are programmable. It helps the applications programmer to better identify problem areas and corresponding diagnostic actions to be taken. It also assists him in reading memory dumps and tracing system control blocks to obtain relevant information. Failing statements in a COBOL program can be located through the use of COBOL/ADE, and it is all done in a conversational mode.

MINIBASIC is a "code and go" implementation of the BASIC language processor. It is fully interactive and generates diagnostics during both compilation and execution.

HARDWARE/SOFTWARE REQUIREMENTS: An average ROSCOE system can be installed on any IBM System/360, 370, 303X, 4300, or compatible computer with at least 512K bytes of main storage, operating under OS, VS1, SVS, or MVS. A typical 6-terminal IBM 3270 ROSCOE system occupies 150K bytes in a main memory partition or region including OS control requirements, but exclusive of syntax checkers. The syntax checkers require anywhere from 6K to 12K bytes each. The UTILITY subsystem requires about 30K bytes, which can be shared with the syntax checkers. Each additional terminal requires about 2K bytes of main memory for local areas and buffers.

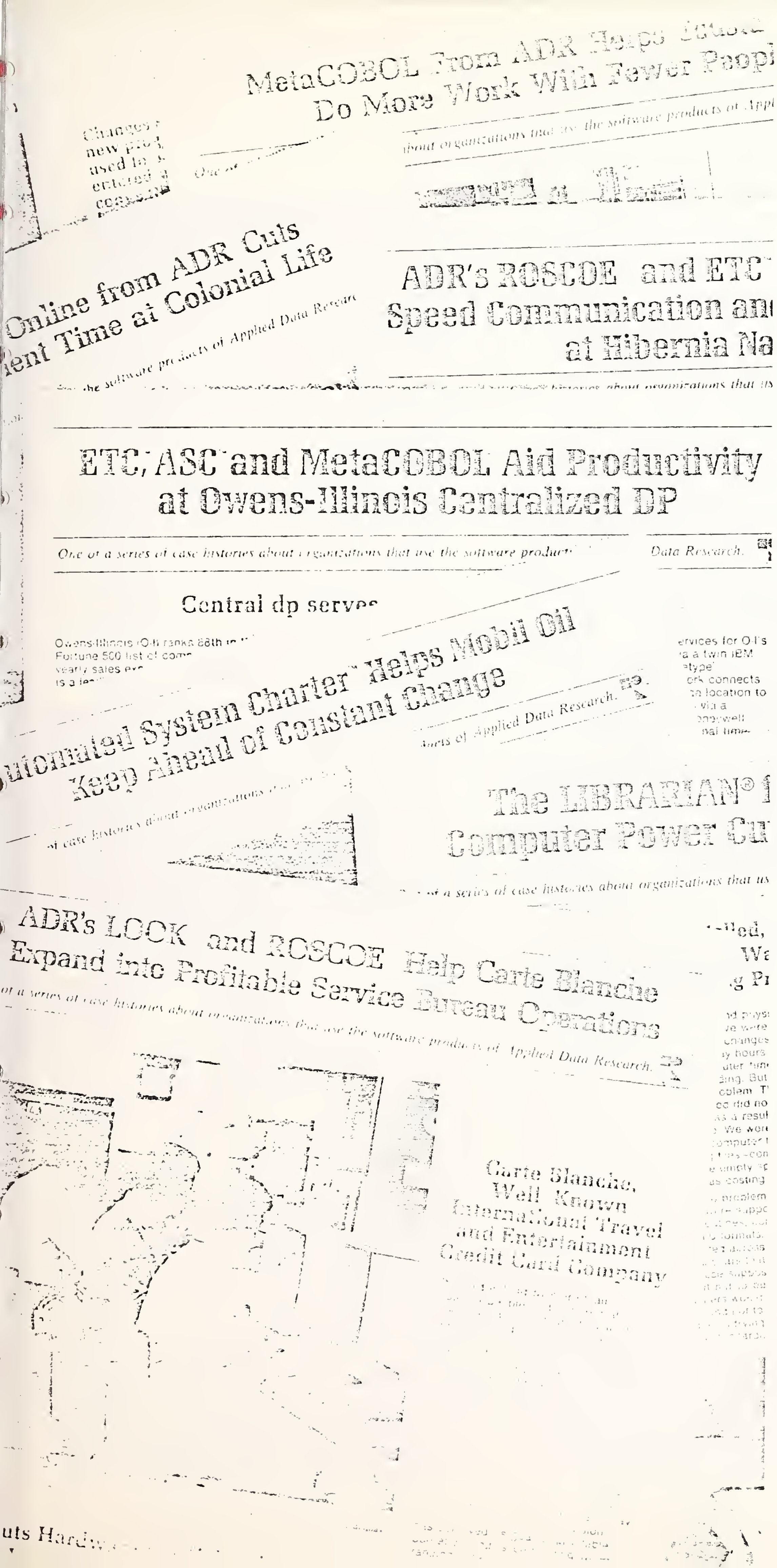
ROSCOE handles terminals capable of running either BTAM or VTAM. It supports such hard-copy devices as the IBM 2741 or Teletype 33/35 as well as locally attached CRT display terminals such as the IBM 3270. Both local and remote 3270's with printers are supported by ROSCOE.

PRICING: ROSCOE and its options are available on permanent license with a single payment, 3 annual payments, or 12, 24, or 36 monthly payments. One year of maintenance is included at no extra charge, and subsequent maintenance is available on 1, 2, or 3-year terms. Long-term leases are also available.

Some of the basic prices are:

	Permanent License	Monthly License
ROSCOE·MVS	$33,000	$1,840
ROSCOE	29,000	1,610
RTS	7,400	410
COBOL ADE	5,000	280
MINIBASIC	4,800	260

■


MetaCOBOL From ADR Helps
Do More Work With Fewer Peopl

about organizations that use the software products of Appl

Changes
new pro
used
entered

Online from ADR Cuts
ient Time at Colonial Life

the software products of Applied Data Resear

ADR's ROSCOE and ETC
Speed Communication an
at Hibernia Na

ETC, ASC and MetaCOBOL Aid Productivity
at Owens-Illinois Centralized DP

One of a series of case histories about organizations that use the software product Data Research.

Central dp serve

Owens-Illinois (O-I) ranks 88th in
Fortune 500 list of com
yearly sales ex
is a lea

ervices for O-I's
a a twin IBM
ctype
ork connects
to location to
via a
neywell
nal time

utomated System Charter Helps Mobil Oil
Keep Ahead of Constant Change

of case histories about organizations

ries of Applied Data Research.

The LIBRARIAN® f
Computer Power Cu

of a series of case histories about organizations that us

ADR's LOOK and ROSCOE Help Carte Blanche
Expand into Profitable Service Bureau Operations

of a series of case histories about organizations that use the software products of Applied Data Research.

Carte Blanche,
Well Known
International Travel
and Entertainment
Credit Card Company

uts Hardw

PRODUCTIVITY

ADR's ROSCOE, VOLLIE, The LIBRARIAN/Online, and MetaCOBOL are improving program and programming productivity through online program development facilities, testing and debugging aids, security and auditing tools, structured programming, and code-generation programs.

OPERATIONAL PRODUCTIVITY

ADR's LOOK, ROSCOE, and The LIBRARIAN are maximizing the effective use of computer hardware and increasing the productivity of operations personnel. These systems are reducing the cost of hardware by providing greater efficiency in the use of existing resources.

MAINTENANCE AND ENHANCEMENT PRODUCTIVITY

ADR's MetaCOBOL and The LIBRARIAN are reducing the costs and increasing the reliability of the billions of dollars worth of programs that require continued maintenance, enhancement, and conversion. As much as 80% of all programming activity is typically directed toward this area alone.

DOCUMENTATION AND COMMUNICATIONS PRODUCTIVITY

ADR's On-line ETC is automating the production of documents, manuals, and other communications materials. Automated System Charter (ASC) and AUTOFLOW II are providing the practical tools for producing and maintaining system-and program-level specifications and documentation. These products provide important communications links between technical and non-technical personnel.

PRODUCTIVITY

ADR's tools for online program development give programmers direct interaction with the computer — to save time, increase productivity, and improve the quality of programs

ROSCOE

A remote job entry system for OS and VS environments that provides conversational facilities for program preparation, testing, and maintenance. Originally developed to provide programmers with an alternative to preparing programs at keypunches and submitting and retrieving their jobs in batch, ROSCOE now serves both application and system programmers, as well as non-programmers, in many phases of their daily operations. ROSCOE with COBOL/ADE™ (the COBOL Automated Debugging Environment) can further assist COBOL application programmers in verifying and debugging their programs in a conversational environment. Installed at more than 500 sites throughout the world, ROSCOE is providing users with an economical, balanced alternative to IBM's TSO and VM/CMS.

VOLLIE

ADR's comprehensive online program development system operational under CICS DOS/VS. VOLLIE offers a sophisticated interface to POWER/VS; complete RJE and RJO facilities; immediate updates to The LIBRARIAN and to the DOS/VS source statement and procedure libraries; Syntax Checkers for COBOL and JCL; and VOICE, the Interactive Command Executor.

VOLLIE's advanced Online Library (the OLL), designed for optimum efficiency in an online environment, provides each user with an almost unlimited permanent and secure data storage area. Data in the OLL are protected from unauthorized access; system functions can be restricted on a user-by-user basis by site management.

"... our productivity has increased, our programming effort is more efficient, and our work is of a higher quality — all because of ROSCOE."
Alexander Metz, Allianz Insurance Co.

"After comparing cost, productivity gains, and CPU overhead requirements, we chose [illegible]."
[illegible]

Today, online programming systems are dramatically increasing programming productivity — not only by decreasing turnaround time, but by providing many psychological benefits as well. ROSCOE provides such benefits for OS environments. Because of its efficient mode of operation, remarkably low overhead, and ease-of-use by both programmers and non-programmers, ROSCOE is winning worldwide acceptance as **the** system for online conversational program development.

Until very recently, Germany's Allianz Insurance Company, the largest insurance company in Europe, was using conventional batch procedures for writing and maintaining programs. Allianz realized it would be more efficient if their 100 programmers could have direct online access to the computer. Therefore, the company organized a team of their DP experts to evaluate ADR's ROSCOE and the other leading time-sharing terminal system.

Alexander Metz, Allianz's Chief Systems Programmer, states that, "After a careful comparison, we found ROSCOE best. ROSCOE is a system with a simple language, good data set organization, data security — and a small CPU load.

"One of the big advantages of ROSCOE is that our programmers can now see the results of their efforts immediately. They enjoy their work so much now that more work gets accomplished. After we installed ROSCOE, the number of jobs that our programmers ran rose from 400 to 800 per day. The overall result has been that our productivity has increased, our programming effort is more efficient, and our work is of a higher quality — all because of ROSCOE. It's beautiful!"

When Carte Blanche, the well-known international travel and entertainment credit card company, expanded its Information Services Division in 1976, it required a programming system that would allow their programmers to work efficiently at remote client locations. Explains Kenneth Panzarella, Information Services Division Vice President, **"We looked at ROSCOE as well as competing systems. After comparing cost, productivity gains, and CPU overhead requirements, we chose ROSCOE.**

"ROSCOE's online programming capabilities allow our programmers to work at peak efficiency regardless of location. Our programmers are happier and they have a higher degree of professionalism. ROSCOE keeps costs down and makes us more competitive. In fact, ROSCOE has increased our ability to do a better job for both our service bureau customers and Carte Blanche. We're positive we made the right choice."

One of the three largest banks in the New Orleans area — Hibernia National Bank — prides itself on its ability to provide highly personalized services to its customers despite rapidly increasing paperwork costs. To increase its programming productivity, Hibernia installed ROSCOE. Don Clark, Data Processing Division Vice President, discusses the reasons why. **"Our 30 programmers and analysts are too highly paid to be spending time keypunching and running around trying to find program decks. To cut that time and get out of the card business, we needed the ability to interact directly with our computer. The logical solution was ROSCOE. Within a week after ROSCOE was installed, we could see the change. The individual who had been in a laborious, archaic card environment had his own online terminal. In effect, ROSCOE gave him his own computer — for online maintenance, development programming, and remote testing. In other words, ROSCOE helps out in nearly every programming function. Everybody wants to use ROSCOE now."**

ADR's VOLLIE is a comprehensive system for online program development in a CICS DOS/VS environment. In addition to complete RJE/RJO facilities and a sophisticated interface to POWER, VOLLIE's powerful set of editing commands provides programmers with efficient tools for program development and maintenance.

Hollywood Federal Savings and Loan of Hollywood, Florida, has found VOLLIE to be invaluable in the continuing development of their banking system. T.V. Jarratt, Vice President of Data Processing, notes that, **"VOLLIE gives us data entry and editing facilities which increase our productivity.**

"For example, when one of our online systems blew up, we didn't know which program in the system caused it, because some of the pointers were destroyed. To diagnose the problem, we displayed the programs that had been running and used the SCAN

command to find a message we had seen in the dump. By scanning for that message, we were able to find the two bad programs out of 200 programs in the system. It would have taken us hours to sit down with a listing and find that message."

"Other editing features help us too," adds John Gregory, Systems Programmer. **"We can duplicate sections of code within a module any number of times without errors because if we have it right the first time, it will be right every time.**

"Even when we scan through 1000 statements, VOLLIE's response time is 2-3 seconds. Any other applications that we have would take 2-3 minutes. And VOLLIE doesn't degrade anything else in the system. ADR must be using some really good techniques in order to get VOLLIE to perform the way it does."

"The individual w
been in a labor
archaic card en
ment now had h
online termina
effect, ROSCOE
him his own com
Don Clark, Hib
National Ba

"ADR must be
ome really goo
niques in orde
get VOLLIE to p
the way it does.
[illegible] Holly
[illegible] Savings

)R's software tools for COBOL de generation and online source ogram management allow the velopment of programs in a orter time and, in the long run, oduce programs which will require ss maintenance

etaCOBOL

)R's COBOL-oriented software prod- t for developing and maintaining ograms at either the conventional)BOL or higher level. MetaCOBOL now in use at more than 450 sites oughout the world for structured ogramming, for code generation, conversion from one COBOL lect to another, for natural inter- ing to a data base management stem, for auditing for adherence to al programming standards, for stematic testing, and for dozens of ner applications. MetaCOBOL is ing countless hours of program- ng effort while making dramatic im- ovements in overall program quality.

brarian

)R's low-overhead, online source anagement system designed for use h The LIBRARIAN at OS, DOS, and installations. LIB/OL is operational der IBM and non-IBM telepro- sing monitors including CICS, TSO, SK/MASTER, DATACOM, and ESTI. LIB/OL allows the display and mediate update of programs stored a LIBRARIAN master file. LIB/OL o provides extensive editing and E/RJO facilities, and the ability display the status of all jobs the system.

"If it were not for MetaCOBOL, our efforts to develop a massive online banking system would have been incalculably more difficult." Peter Aebischer, Swiss Bank Corporation

"To keep costs down, we needed a way to shorten the process of coding COBOL programs. MetaCOBOL met that need." George Shaw, Owens-Illinois

The number of debugged program statements produced with a given effort over a given period of time is the ultimate measure of program development productivity. A COBOL preprocessor which can amplify the benefits of a COBOL compiler (just as the assembler and its successor, the compiler, amplified the original machine code concept) can result in a many-fold increase in program development productivity.

MetaCOBOL is such a preprocessor. Unlike any other COBOL programming aid, MetaCOBOL also offers the unlimited power of a true macro language. With MetaCOBOL, users can extend COBOL for nearly any purpose while virtually ending tedious and repetitive work.

The Swiss Bank Corporation, headquartered in Basel, Switzerland, recently installed a sophisticated online system to reorganize its daily handling of thousands of banking transactions. The development of this system required a massive effort (almost 280 man-years). If it were not for ADR's MetaCOBOL, however, these efforts would have been incalculably more difficult. According to Peter Aebischer, Manager of Systems Development, **"We have built up a library of 145 macros which has helped us introduce hundreds of programs in a very short time period. MetaCOBOL has greatly reduced programming time because we are able to write a typical transaction with about four pages of MetaCOBOL macro coding, which is then automatically expanded to about 40 pages of COBOL coding.**

"If you compare only the coding and verification for such a program, it would take a month to develop it in normal COBOL code. It takes only one day with MetaCOBOL — a time reduction of 20 to 1."

Similarly, Owens-Illinois, whose central DP headquarters in Toledo serves 115 plants and offices throughout the U.S., reported a 15-20% jump in programming productivity by using MetaCOBOL macros as shortcuts to enter programming commands.

George Shaw, Internal Project Coordinator, explains. "When O-I took a serious look at programming productivity and format standards, one thing we needed was a way to shorten the process of coding COBOL programs. MetaCOBOL met that need. A programmer can enter a short, 3-line MetaCOBOL macro command, for example, and have it expanded to 50 lines of error-free COBOL source code every time. Now, practically all our COBOL programmers use MetaCOBOL facilities. As a result, productivity has increased dramatically."

"MetaCOBOL is a tremendous tool which allows us to get things done with less actual coding," added O-I's Special Projects Supervisor, Howard Abernathy. "We now wonder how we ever got along without it."

MetaCOBOL facilities are also used to raise the level of COBOL by adding the "constructs" of structured programming. Sweden's $2.8 billion Kooperativa Forbundet (KF), one of the largest retail organizations in Europe, implemented this technique to bring about major increases in productivity. Michael Evans, KF's EDP consultant, comments, "One of our data processing goals has been to keep the number of programmers constant over the years, in spite of greatly increasing workloads. To reach this goal, it was clear that we had to enable programmers to work with more code. When we adopted structured programming, we chose ADR's MetaCOBOL as the tool to make it work. If we had implemented structured programming without using MetaCOBOL, the COBOL code we produced would not be clear.

"With MetaCOBOL, the structure is in the code. Our programmers know immediately what is happening without superfluous comments. And, when the programmers maintain their programs, they have automatically documented the changes because the documentation is all right there as part of the code.

"Today our programmers have no doubt that MetaCOBOL's structured programming language is the only way to write programs. They make fewer mistakes, they spend less time on maintenance, and they can handle a larger workload."

The LIBRARIAN/Online (LIB/OL) has brought the benefits of online source program management to users of The LIBRARIAN. LIB/OL users can display and immediately update programs stored on a LIBRARIAN master file and can submit jobs and review the output at the 3270 terminal. LIB/OL's scratch pad file provides a full-screen edit capability for efficient program modifications.

Such facilities have given Colonial Life Insurance Company of East Orange, N.J., a number of time-saving advantages. As Howard Lackow, Vice President for Data Processing, says, "In the first six months of operation, we have seen a steady increase in the number of lines of source code written with the same size staff. Although we had been steadily increasing the number of lines of code written each month for some time, after we installed LIB/OL the number of lines written jumped by a factor of three in the first four months of use. We're now writing over 40,000 lines a month and still increasing in very significant jumps.

"We have also seen that LIB/OL has given our programmers a lot more control. Because a programmer no longer has time lapses between parts of a job (as he had with batch operation), he doesn't have to reorient himself all the time in order to continue a job. He now concentrates on one program until all the compilation errors are corrected and it is ready to test. It has meant a real increase in productivity and in programmer morale. Programmers are happier now because they can start jobs and see finished results in one session."

Greg Walsh, Assistant Vice President, adds that, "We have run reports when managers wanted to see the bottom line without getting all the intermediate data. Remote Job Output has let us display the last page of a report, get the information we want and then flush the job — before it ever gets to the printer. We found too that the Remote Job Entry portion of LIB/OL is not restricted to program maintenance for our source code. With RJE we can enter the JCL to the computer to process any job. We use the scratch pad area to assemble the JCL to run a complete job stream and then submit that as a job through The LIBRARIAN facilities. It's not restricted to program development type work. It's any remote job entry."

"After we installe
LIB/OL, the number
program lines writt
jumped by a factor
three in the first fc
months of use." How
Lackow, Colonial L
Insurance Compan

"With MetaCOBOL,
structure is in the c
If we had implemen
structured programi
without using MetaCC
the COBOL code we
duced would not be cl
Mike Evans, Kooper
Forbundet

R's LOOK, ROSCOE, and
e LIBRARIAN help your
nputer keep pace with growing
a processing requirements

OOK

nique real-time measurement
tem essential to any IBM/360 or
0 installation interested in im-
ving performance of its MVS, VS1,
S, or MVT environment. LOOK is a
ible, wide-ranging tool capable of
viding an immediate and accurate
raisal of today's dynamic com-
er environments.

SCOE

emote job entry system for OS
ironments that provides a conver-
ional approach to program
paration, testing, and maintenance
remote terminal facilities.
SCOE has a significant impact in
gram development and main-
ence, system programming ac-
ties, production coordination and
eduling, interactive problem
ving, and DOS-to-OS conversions.

brarian

comprehensive source program
nagement system offering sophis-
ated data storage and manipulation
ilities for programming, systems,
d operations personnel and their
nagers. Along with its facilities for
tralized program storage and pro-
m security, The LIBRARIAN makes
remely efficient use of data stor-
facilities; COBOL programs stored
a LIBRARIAN master file typically
be compressed up to 75%.

"The way LOOK is implemented...is a brilliant solution."
Arne Dale,
Rogalandsdata

[illegible] see where [illegible] are [illegible] immediate [illegible]

Many modern DP installations are plagued by constant demands for acquisition of more system resources — more memory, more disk storage, even more CPUs — to meet the ever-increasing needs of their users. Quite often, however, these needs can be satisfied through better use of existing hardware.

LOOK makes more resources available for productive work by providing instantaneous information about common operational conflicts — circumstances that quickly degrade throughput and slow a production system.

LOOK not only provides information on resource use and performance, but also provides the controls to dynamically correct system imbalances. All this with only a 1% overhead — 1/10 that of most other performance monitors.

LOOK is a system that is now keeping hundreds of installations fine-tuned and operating more efficiently.

Rogalandsdata, a modern data processing center in Stavanger, Norway, found this to be the case when it was running its operation on their 370/145. Explains Arne Dale, Rogalandsdata's Software Manager, "It was a strange situation! The system was completely overloaded, the CPU usage was 90-100%, and the paging rate was around 20 pages per second during the entire day shift. LOOK helped us get better throughput.

"When we first considered LOOK, we were very skeptical because we didn't see how it could get such reliable data using so little of the system's resources. But we checked LOOK against other measurements and found that its data was accurate within a few percent.

"The way that LOOK is implemented, using variable sampling and as an extension to the OS operator commands, is a brilliant solution."

Carte Blanche used LOOK to increase the efficiency of its hardware and to prevent lost computer time. Kenneth J. Panzarella, Information Services Division Vice President, describes their situation. "In the service bureau business, lost time quickly means lost revenue. To prevent that, our operations group uses LOOK throughout the day, often on a minute-to-minute basis, to see what is in the computer, what parts are being used, and what jobs are waiting to be processed. LOOK lets us see where the inefficiencies are and take immediate corrective action."

"LOOK is a powerful tool for many special situations, but they are the types of situations that are with us on a day-to-day basis. As a result, LOOK works for us continuously," added Software Analyst, Bob Dunsire.

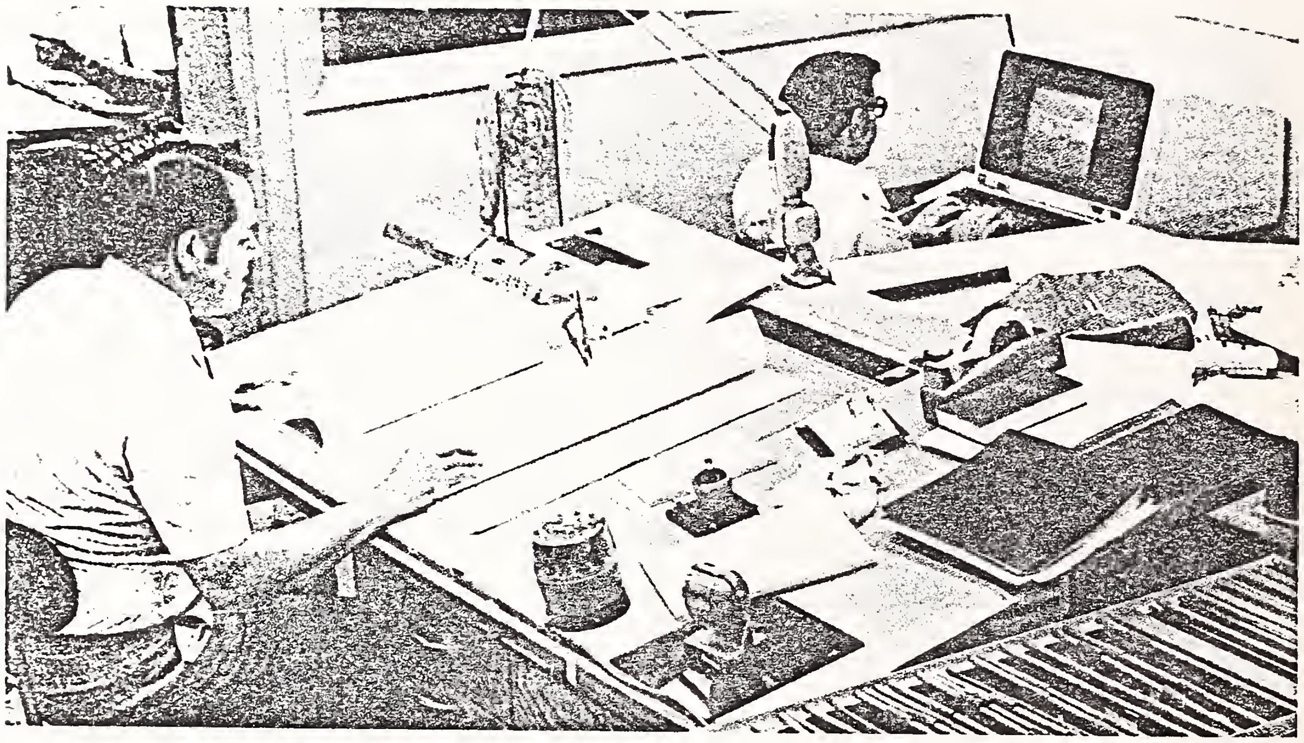
Online program development plays a major role in increasing programmer productivity. Often, however, the addition of an online system can create a need for expensive hardware which many companies feel they cannot afford. ADR's ROSCOE combines the benefits of online conversational programming with operational efficiency.

Before Allianz Insurance Company installed ROSCOE, for example, it made a careful comparison of leading terminal systems. Chief Systems Programmer Alexander Metz, who helped perform the evaluation, explains the reasons for their choice. **"Because our CPU was already being utilized 80-100% by teleprocessing and batch jobs and because we could not expand our hardware, our most important criteria for the choice of a terminal system were performance and CPU load. After a careful comparison of leading systems, we found ROSCOE best. During our SVS test evaluation period, for example, we found that ROSCOE's overhead was under 1% per terminal, while the other system we evaluated required up to 4.5% per terminal.**

"At our headquarters in Munich, 110 users now have access to ROSCOE through 45 terminals — for systems and application programming, job preparation, information retrieval, project planning, monitoring, and documentation. I don't believe there is any other system like ROSCOE that can serve 45 terminals with such good response time."

"Our most impo
criteria were per
ance and CPU loac
a careful compari
leading systems
found ROSCOE h
Alexander Me
Allianz Insurar
Company

"I don't believe t
any other syste
ROSCOE that ca
45 terminals wit
good response t
Alexander M
Allianz Insura
Company



'ith ROSCOE/RTS,
.chine overhead was
bout 70% less than
with TSO."
Bob Wells,
North Carolina
.ate Highway Dept.

Similarly, the North Carolina State Highway Department wanted to install an online personal computing system for its highway engineers. Its first thought was IBM's TSO. Already shackled with heavy CICS applications, the addition of TSO would have required immediate and expensive upgrading of their 370/158.

That's when the Department of Transportation heard about ROSCOE's Time-Sharing Option (RTS). Computer Analyst Bob Wells outlines his reasons for selecting ROSCOE/RTS. **"ADR let us put up ROSCOE/RTS for a 30-day evaluation and, at the end of that time, it was clear it would do the job we needed. Some of the important factors in our decision were:**

- **With ROSCOE/RTS, we could continue with VS1 and not have to go to MVS.**
- **Machine overhead was about 70% less than with TSO.**
- **We were familiar with ADR through the use of The LIBRARIAN and knew that if we had problems, we could talk directly to the people who developed the software."**

Conservation of valuable resources is characteristic of ADR's software products. And conservation in one area can have a significant impact in other areas as well. For example, when ADR's The LIBRARIAN was installed at Computer Power Inc., a Jacksonville, Florida, mortgage banking computer service center, the entire source program

library was immediately compressed onto two disk packs online on two separate drives. Because it took full advantage of common characteristics among source languages and because it never allowed random empty spaces to accumulate, The LIBRARIAN eliminated the use of 15 disk packs.

Today, Computer Power has switched to the IBM 3330 disk system. Their entire source program library, consisting of 8400 program modules, is now stored on a single online disk. As a result, they are saving nearly 50 hours a month of vitally needed computer time because they have no mounts/dismounts to perform.

PRODUCTIVITY

ADR's software products reduce the cost and time required to maintain, enhance, and convert programs

MetaCOBOL®

A multi-purpose COBOL-oriented software product designed to assist in coding, testing, debugging, standardizing, converting, and optimizing COBOL programs. Users have found that MetaCOBOL has accounted for a dramatic reduction in the cost of maintenance and enhancement and, in many cases, has enabled them to perform conversions of many programs more quickly.

Librarian®

ADR's system for source program retrieval and maintenance is one of the computer industry's most widely installed products with over 5000 installations worldwide. Its facilities for updating programs and maintaining a complete history of program changes have resulted in major reductions in the total maintenance effort.

"MetaCOBOL lets us track down different programming styles and get into the problems much more quickly."
Nick Kaluger,
Ecusta Paper

"MetaCOBOL has been a tremendous help in cleaning up and standardizing programs that were written over the years."
Nick Kaluger,
Ecusta Paper

Many companies report that the large majority of their programmers' time is spent maintaining and enhancing existing applications. Not only is time and effort great, but companies also are frequently faced with an added problem: individual programmers whose unique styles of program construction make it imperative that the original programmer performs maintenance.

ADR's MetaCOBOL substantially reduces these problems by automatically enforcing programming standards and providing extensive facilities for program upgrading and optimization. MetaCOBOL makes it possible for almost any programmer to change and upgrade anyone else's programs with a minimum of effort.

Ecusta Paper Division of the Fine Paper and Film Group of Olin in Pisgah Forest, N.C., has found MetaCOBOL to be a useful tool in upgrading and improving a large inventory of programs, many of which were written before

today's COBOL standards came into widespread use. According to Nick Kaluger, Manager of Systems Development, "We bought ADR's MetaCOBOL mainly as a tool to clean up and standardize programs that were written over the years. These programs were assets, just like our plant building or paper- and cellophane-making machinery. These were assets that we thought could be salvaged. They were. And MetaCOBOL has been a tremendous help in that effort.

"MetaCOBOL helps the programmer get into a program faster. It helps us enforce details like alignment, paragraph numbering, level numbering, common names, etc.

e have more time
rk in new areas, . . .
e can introduce ap-
ations which would
been uneconomical
few years ago."
eter Aebischer,
s Bank Corporation

nks to MetaCOBOL,
one can now easily
ntain programs or
changes, no matter
o the author was."
eter Aebischer,
Bank Corporation

Until recently we had various styles of programs. Each programmer would have his own little pattern. Some would do repeats at the bottom, others would do them in the middle. Some had little loops, others had giant loops. Things like that are still in the programs, but we are now able to track down these different styles and get into the problems much more quickly."

The Swiss Bank Corporation has found that MetaCOBOL has had a large impact on maintaining programs for their real-time banking system. As Peter Aebischer, Manager of Systems Development, explains, "Before we used MetaCOBOL and its macro facilities, programs were essentially the 'property' of the individual programmer who wrote them. No one else could easily follow his work. If a program needed maintenance, the original programmer had to be there to do it. If he were on vacation and if maintenance were urgent, another programmer had to go into the code and spend perhaps three weeks just making a small addition — usually finishing just a few days before the original programmer returned anyway.

"The concept of individual program authorship is now all changed. Thanks to MetaCOBOL, anyone can now easily maintain programs or make changes, no matter who the author was. Even a supervisor can maintain programs when no one else is available because now he has access to the needed information in seconds.

"Our programmers are relieved and happy because they can spend most of their time doing what they like to do best — writing new programs, new applications. And because we have more time to work in new areas, our costs are down and we can introduce applications which would have been uneconomical a few years ago."

Leeds & Northrup Corporation, a major manufacturer of process control instrumentation, found that MetaCOBOL played a key part when they converted and upgraded more than 1500 programs. In planning for the OS system which is now installed, Leeds & Northrup was required to convert from DOS ANS COBOL to OS ANS COBOL. MetaCOBOL was the tool that made this possible.

MetaCOBOL was installed and 80% of the conversion to OS ANS was completed in less than three months.

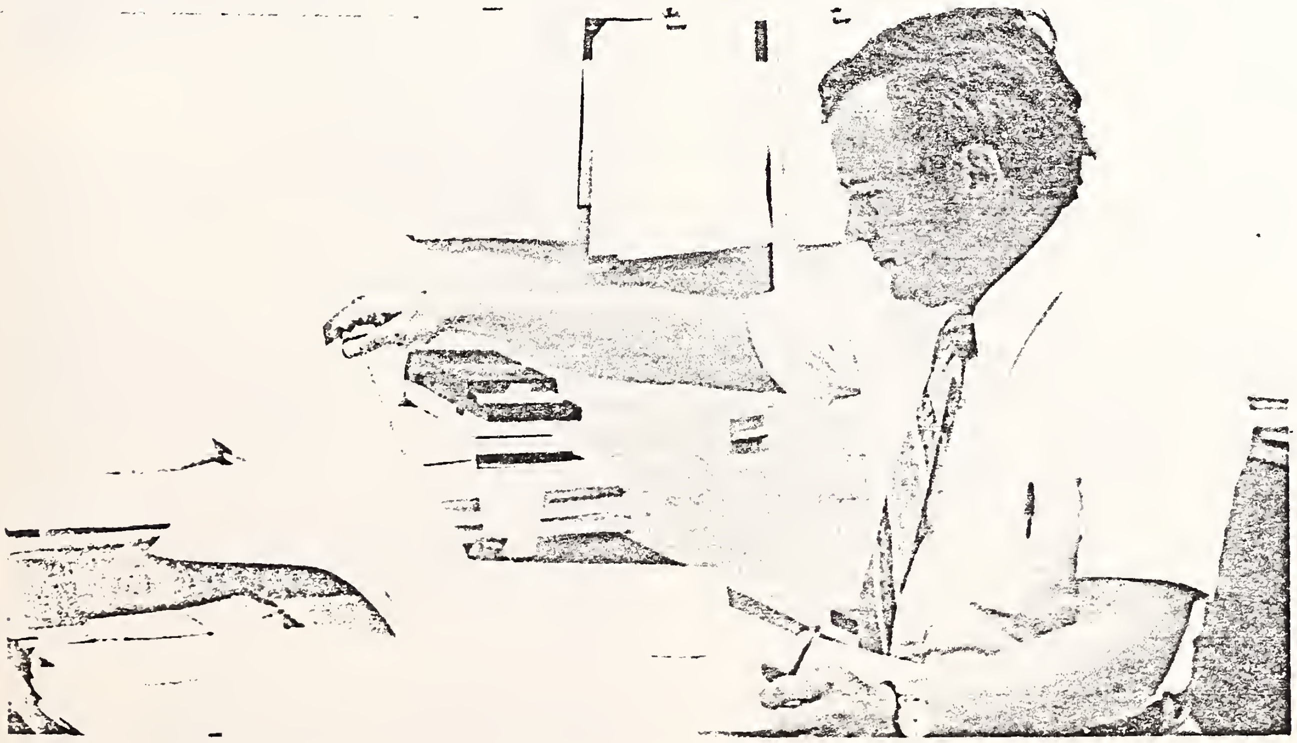
According to Joe Dupon, Manager of Computer Technology, **"We figured with MetaCOBOL that maybe 10% of the programs would have to have some manual intervention in the code conversion. We were happily surprised to find out that we had to touch less than 1%. We reached a point in the conversion cycle when we took DOS programs, fed them into MetaCOBOL, and took them through the compiler, link-edited them, and had them ready to go. All we had to do then was create the conversion JCL."**

MetaCOBOL conv
80% of Leeds
Northrup's progra
less than three mo

ADR's The LIBRARIAN was initially installed four years ago at Pepsi-Cola to simplify and control program update procedures. It is now a fundamental tool used to maintain central control over the use and modification of all programs. Its facilities for source program maintenance, control, and security have made strict accountability an almost automatic daily process.

The LIBRARIAN
facilities for sou
program mainten
control, and secu
have made stri
accountability an
automatic daily pr
at Pepsi-Cola

Over 200,000 man-hours are invested each year in software system development, documentation, management, and use. Pepsi-Cola looks upon its program library as an important asset to be protected and made more efficient in every possible way. And software products from Applied Data Research are an important part of that effort.



D COMMUNICATIONS
RODUCTIVITY

's software products provide
means to generate and update
ual information, system
cifications, and program
umentation

line ETC

nline word-processing tool
h utilizes computer resources to
mate the preparation, and
tenance of any textual material.
loped in response to the many
comings of most manual (and
the more sophisticated word-
essing) approaches to document
uction, ETC's automated facil-
expedite even the most tedious
ents of document preparation —
ing, editing, final typing, and
ing.

AUTOMATED SYSTEM CHARTER

mary system documentation tool
reating and maintaining diverse
of system- and application-level
mation. The various high-level
s and reports automatically pro-
d by ASC create a reliable and
sive network of information for
ing systems, as well as new
ms under development.

II

dvanced system development
which analyzes source module
and automatically produces
sive cross-referenced listings,
ic charts, and detailed sum-
s of individual program activity.
management tool, AUTOFLOW II
ts in creating and sustaining a
olled and organized installation
onment: as a programming tool,
FLOW II helps in developing
naintaining accurate and reliable
cations.

"Before ETC, it took forever to get a manual redone. Now new updates appear in a matter of days." Reg Campbell, Hibernia National Bank

"At Owens-Illinois, 15 procedures manuals containing about 10,000 pages are maintained through ETC by a single clerk," George Shaw, Owens-Illinois

In today's competitive corporate environments, good documentation is essential to an efficient operation — not only documentation for use within the data processing environment, but also manuals and directories used throughout an organization. Producing and updating such documents, however, often strain an organization's resources to the point that needed documentation is either not produced or, at best, is badly out-of-date.

Recognizing these problems, ADR has developed tools to harness the computer to generate and maintain documentation in a fraction of the time normally required by manual methods.

ADR's On-line ETC (Extended Text Compositor) is a word-processing system designed to facilitate the production of a wide range of textual material — from highly technical reference manuals to telephone directories, form letters, and memoranda. ETC enables terminal operators with minimal training to economically generate and maintain thousands of pages of material.

New Orleans' Hibernia National Bank has found that ETC not only reduces the time and effort they spend to produce a manual, but also allows them to keep manuals up-to-date and to improve communications within their branches as well. Reg Campbell, Systems and Programming Manager, comments that, "Our manuals are used by personnel throughout our entire banking system. Anyone who prepares any type of data for input to the computer is guided by one or more of our procedures manuals.

"Since our personnel heard about our use of ETC and how easily changes are made, they are no longer hesitant to suggest improvements or modifications to procedures. Before ETC, it took forever to get a manual redone. Now, however, their suggestions are likely to appear in a new update in a matter of days. ETC has really helped to improve communication with the people who have to use our manuals in their day-to-day work.

"A tool as powerful and flexible as ETC impressed me because I could train somebody so easily. When new secretaries come into the department, they practice with the terminal and it becomes a challenge. They really like it."

"At Owens-Illinois, 15 procedures manuals containing about 10,000 pages are maintained through ETC by a single clerk," explains Internal Coordinator George Shaw. **"But only about a third of her time is spent in keeping up with all the changes to our procedures manuals. ETC lets us create a perfectly formatted document without having to go to a typist. Now, our typists are primarily concerned with short administrative letters since all documentation is handled by ETC.**

"With ETC, we can change a page here and a section there and get the updated information to the right people without having to redo the entire manual — and it's inexpensive. ETC is a powerful piece of software. And when you consider the cost of copying and the cost of delays that come with hand-typing, it becomes a tremendously worthwhile package."

ADR's Automated System Charter (ASC) is a dynamic remedy for the pressing need for effective system documentation. It permits the user to easily obtain the critically needed information which is essential to a disciplined and productive approach to system maintenance and enhancement. The high-level system charts and reports automatically produced by ASC create a reliable and extensive information network.

Mobil Oil's New York DP headquarters uses ASC to keep up with changes to over 300 systems, a third of which may be extensively changed in any year. Maintaining up-to-date standardized documentation in this environment by manual methods would be quite impossible. According to Jean-Louis Legrand, Mobil's Client Services Supervisor, **"Before ASC, there was no standard way of presenting system charts. Some programmers would use squares and others would use circles to describe the same type of step. Some charts were very detailed, and some were not detailed at all. Each chart ended up reflecting many individual styles and preferences. Now the results are more uniform and complete, and it is much easier for any of our programmers or Client Coordinators to quickly understand a system.**

"It is difficult to estimate costs when you are dealing with people's time, especially programmers writing documentation. But, by our most reasonable estimates, we found that ASC has enabled us to document a system in about half the man-hours that we would have normally used."

ADR's AUTOFLOW II and Cross-Program Auditor (CPA) help create and maintain the disciplined, well-structured programming environment essential to the effective management of program development and maintenance.

AUTOFLOW II accepts and analyzes over 20 source languages to produce extensive cross-reference listings, summaries, and graphic charts of various aspects of program or module activity and logic.

AUTOFLOW II's predecessor, AUTOFLOW, is one of the first software programs ever to be granted a U.S. patent. AUTOFLOW II today represents nearly ten years of software design and experience focused on increasing efficiency, economy, and effectiveness in programming operations worldwide.

CPA, available under AUTOFLOW II or The LIBRARIAN, examines and reports upon characteristics of an entire system of COBOL programs, across individual program boundaries, within the context of their functional interaction.

**"ETC lets us crea
perfectly format
document without h
to go to a typis
George Shaw
Owens-Illinoi**

**"ASC has enable
document a syst
about half the ma
that we would
normally use
Jean-Louis Leg
Mobil Oil**

"An important point about ADR's software products — ROSCOE, The LIBRARIAN, and MetaCOBOL — is that they have always worked to specifications.

"The program packages that have been shipped to us have had a virtual zero defect level. As a result, we have never hesitated to put them to work in our system without the lengthy trial and test procedures that we normally use with our other products."

Joe Dupon, Leeds & Northrup

"Our experience with VOLLIE has been indicative of the attitude ADR has for its product. They don't even want to send a system to their beta test sites unless they are pretty well satisfied that it is clean."

John Gregory, Hollywood Federal Savings & Loan.

"At Rogalandsdata, we are committed to providing the best service at a reasonable cost. We consider a number of factors when deciding which software products will be installed for general use: performance, reliability, efficiency, and compatibility with other software already in the system. Rogalandsdata and [illegible] users think that The LIBRARIAN, MetaCOBOL, ROSCOE, and LOOK [illegible] [illegible] standards."

[illegible]

INSTALLATION AND TRAINING

ADR's products are installed throughout the world by ADR-trained technicians. These Technical Support Representatives also train users thoroughly in the use of the products. ADR's systematic approach to installation and training ensures a smooth transition, for both operations personnel and users, between current practices and the new environment.

PRODUCTS DELIVERED READY TO WORK

When an ADR product is installed, the user can be sure that it will work to specification. Every ADR product or product enhancement undergoes thorough alpha and beta testing to assure near-perfect product reliability. Only then is a product released to the user community.

DOCUMENTATION

ADR's products are fully documented in manuals which provide all the information necessary for efficient installation and effective use. ADR supports a 'common sense' approach to technical documentation, in which information is presented in simple, direct language and supported liberally with tables, exhibits, and examples. All documentation is thoroughly edited and field-tested and is constantly reviewed by both the editorial and product development staffs. This continual effort ensures that readers at every level will be able to achieve a clear and complete understanding of the operation and capabilities of the product.

SUPPORT

ADR's Technical Support Representatives are available for on-site assistance when needed. ADR's representatives are highly trained specialists capable of providing expert help for the full range of ADR's products in all operating environments. Realizing the critical nature of computer operations throughout the world, ADR's representatives or product specialists are available by telephone or telex whenever an emergency arises.

USER GROUP MEETINGS

Users of ADR's software products meet both nationally and internationally to exchange problem-solving ideas, techniques, and experiences and to learn of enhancements planned for the product. Published proceedings are distributed to further encourage effective and new uses of the products.

WORLDWIDE OFFICES

ADR is the only software company with offices throughout the United States and representatives in more than 40 cities throughout the rest of the world. Each office is staffed with ADR-trained specialists ready to serve your needs.

"We brought in a software product from another company. When we tried to install it, we found problems with it and ended up performing debugging for the vendor. Needless to say, we couldn't use the product on that basis and sent it back. That has never happened to us with ADR products. We've found them to be ready to use as shipped, and they're installed quite easily. And we've had technical assistance from ADR technicians when we needed it."
Gary Hall, Ohio Medical Indemnity

"The ADR ROSCOE User Group is one of many ROSCOE benefits. I went to my first ROSCOE User Group meeting and came back with new ideas and names of people at other companies who had done exciting things with ROSCOE. Because I got so much out of my first meeting, I intend to go to every one so that I can continue to exchange ideas and find new problem solutions."
Bob Dunsire, Carte Blanche

"Whenever we have brought in a new release of LIB/OL, we've been so confident that we have just loaded it right on top of the old one. We do a daily backup of course, but we don't have to do any thorough testing. We just throw a new release out there, and it generally works without difficulty. If we do have any questions, we just call the ADR support group. They've been very helpful. We haven't had any real problems at all."
Gordon Morris, Merchants National Bank

Applied Data Research's computer software products are regarded by their users as Number 1 in the industry according to the results of the recent 1977 Datapro survey of users of software products. ADR won more honors than any other firm, receiving top-rated Honor Roll awards for The LIBRARIAN, LOOK, ROSCOE, and an honorable mention for MetaCOBOL. The survey was based on questionnaires sent to more than 30,000 data processing executives. With more than 5800 respondents commenting on 1223 different packages, only 44 packages received Honor Roll or honorable mention status for overall user satisfaction.

Award-winning recognition is nothing new to ADR. Its products have been cited every year in the Datapro survey, with a total of 11 commendations in the 5-year history of the awards.

Overall user satisfaction with ADR products has been rated in another important way, too — overall product sales. Five of ADR's products have been the recipients of the ICP Million Dollar Awards: The LIBRARIAN — 20 Million Dollar, ROSCOE and AUTOFLOW II — 10 Million Dollar, MetaCOBOL — 5 Million Dollar, and LOOK — 1 Million Dollar Awards.

1973
The LIBRARIAN

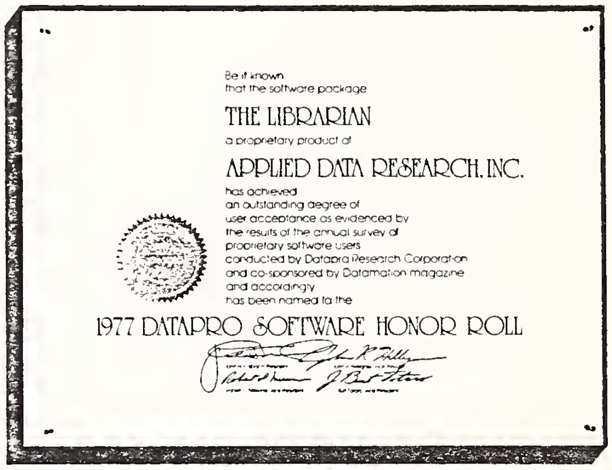
1974
The LIBRARIAN
MetaCOBOL*
PiSort*

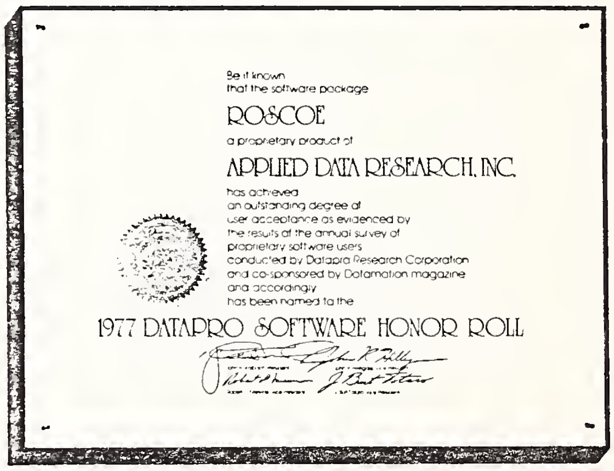
1975
The LIBRARIAN

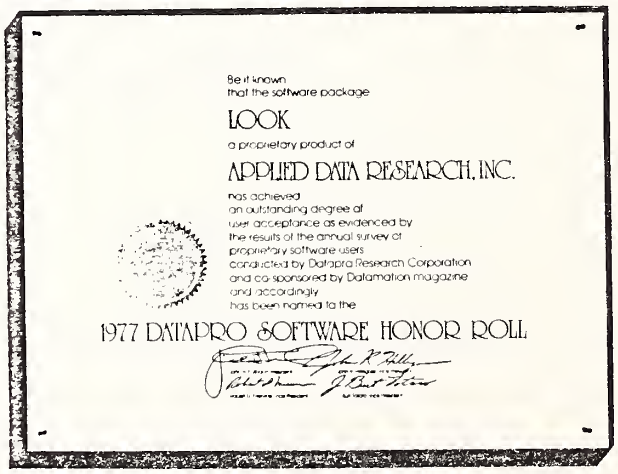
1976
The LIBRARIAN
LOOK

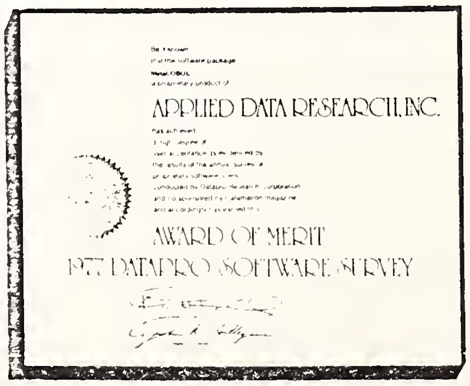
1977
MetaCOBOL*
The LIBRARIAN
LOOK
ROSCOE

*Honorable Mention


Be it known
that the software package
THE LIBRARIAN
a proprietary product of
APPLIED DATA RESEARCH, INC.
has achieved
an outstanding degree of
user acceptance as evidenced by
the results of the annual survey of
proprietary software users
conducted by Datapro Research Corporation
and co-sponsored by Datamation magazine
and accordingly
has been named to the
1977 DATAPRO SOFTWARE HONOR ROLL


Be it known
that the software package
ROSCOE
a proprietary product of
APPLIED DATA RESEARCH, INC.
has achieved
an outstanding degree of
user acceptance as evidenced by
the results of the annual survey of
proprietary software users
conducted by Datapro Research Corporation
and co-sponsored by Datamation magazine
and accordingly
has been named to the
1977 DATAPRO SOFTWARE HONOR ROLL


Be it known
that the software package
LOOK
a proprietary product of
APPLIED DATA RESEARCH, INC.
has achieved
an outstanding degree of
user acceptance as evidenced by
the results of the annual survey of
proprietary software users
conducted by Datapro Research Corporation
and co-sponsored by Datamation magazine
and accordingly
has been named to the
1977 DATAPRO SOFTWARE HONOR ROLL


Be it known
that the software package
MetaCOBOL
a proprietary product of
APPLIED DATA RESEARCH, INC.
has achieved
a high degree of
user acceptance as evidenced by
the results of the annual survey of
proprietary software users
conducted by Datapro Research Corporation
and co-sponsored by Datamation magazine
and accordingly has earned this
AWARD OF MERIT
1977 DATAPRO SOFTWARE SURVEY

INTERACT

INTERACT

Summary Description

Revision 1

Release 6.0

March 1979

Cullinane Corporation

Introduction

INTERACT is an online system that offers an efficient way of handling text entry, text manipulation, and remote job processing. INTERACT is conversational and terminal-oriented. It affords high performance standards, reliability, and considerable cost savings.

With INTERACT, data such as text documents, programs, or JCL are readily available on a current basis for either examination or change. Remote terminals connected to a central computer through INTERACT allow users ready access to their information base. In addition, users may submit jobs to the central computer's job queue from a remote terminal and retrieve job output at that location.

INTERACT provides the following facilities:

- **Text entry and editing.** Text can be entered from a terminal or through conventional batch methods. Once entered, data can be manipulated by means of INTERACT commands. When modifications are made, INTERACT displays the revised text as the changes are applied.

- **External storage.** Text can be saved on disk and instantaneously retrieved at the terminal for examination or updating.

- **Remote job processing.** Jobs can be submitted and monitored from a remote terminal. Job output can be directed to the terminal and portions can be printed or discarded as required.

- **Text and letter processing.** Using INTERACT's AUTOFORM facility, text can be formatted into camera-ready copy of documents, letters or manuals. In conjunction with the letter writer, labels lcan be generated for complete automation of mailings.

- **Interactive programming.** INTERACT commands in combination with the INTERACT preprocessor act as a powerful high-level programming language. The Execute Command Language can be used to perform arithmetic and logical computations as well as many of the other functions associated with a programming language.

- **Accounting facilities.** INTERACT generates records after each session that provide statistical information regarding that session for the System Management Facilities (SMF) data set. These records can be used to produce INTERACT reports that detail individual and group statistics and changes.

The User and INTERACT

Communications at the terminal are user-oriented and conversational. INTERACT communicates with the user during the entire terminal session, responding to requests from the user and asking questions if more information is needed. The commands are easy to learn because they have an English-like structure. INTERACT commands have short forms (abbreviations) that make it easier for the non-typist to use.

While signed on, each user has a one-to-one relationship with the central computer. It appears as if there is only one user on the system. Practically everything the user asks INTERACT to do is done independently from any other processing that may be occurring within the system. A user may submit jobs, send documents to a high-speed printer, and begin revisions on another document — with each task executing concurrently.

In the event of a system or a communication line failure, INTERACT provides full recovery; no data is lost, so time-consuming re-entry is eliminated. Recovery is performed automatically and instantly; as soon as the INTERACT system is restarted after an operating system failure or as soon as a user signs on again after a communication line failure.

INTERACT Computing Applications

Program Development. INTERACT is ideal for programmers. INTERACT increases the number of turnarounds available on a daily basis as well as decreasing the time between turnarounds. INTERACT gives the programmers visibility and control of their own jobs. This not only increases productivity but it greatly increases job satisfaction. Because tasks can be completed sooner, programmers have the option of working on fewer tasks simultaneously, thus reducing the amount of time wasted switching from one task to another.

In addition to better turnaround, the power of INTERACT commands reduces the amount of keying required to perform program maintenance. By using the execute facility, programmers can accomplish some tasks in minutes that would otherwise require days or weeks. For example, an execute program can be created to search an entire library for all occurrences of the word '3330' and list them or change them to '3350'. This capability makes INTERACT an excellent conversion tool.

Timesharing. INTERACT's English-like command structure and the Execute Command Language make it excellent for non-programmers. These users receive all of the benefits of INTERACT's fast turnaround and, at the same time, are not subjected to the problems JCL causes them. Once a set of execute programs has been created for a specific set of users, such as an engineering group, all their work can be done completely without reference to JCL or other computer-oriented problems. This gives users direct control over their work, significantly reducing the amount of support necessary from the data processing group.

Word Processing. INTERACT excels in high volume word processing applications such as letter writing, label generation, and the preparation of large documents. The ability to directly access computer files adds a new dimension to word processing. With its ability to prompt for information, AUTOFORM extends word processing to untrained word processing operators. Prompting also reduces errors and makes it easy to perform applications which are not frequently performed and therefore may not be remembered from one use to the next.

Production Control and Scheduling. Reruns due to improper input are a continual problem in most production control groups. The ability to edit responses given by the operator makes the Execute Command Language ideal for entry of data which must be correct. In addition, online files such as the master schedule file may be accessed for additional editing. Execute programs can be written by the control group — programming experience is not necessary. In fact, many INTERACT installations have automated their production scheduling, resulting in substantial personnel savings as well as a reduction in errors.

INTERACT Facilities

Text Entry and Editing. Text entry operations include automatic prompting and line numbering, tabbing, simple backspace corrections or discarding of entire lines. Powerful text-editing operations allow deleting, inserting, and replacing single characters, character strings, single lines, or line ranges. Lines may be moved or copied without renumbering the entire document. Ranges may be defined by line number, column number, string occurrence or the absence of a string. Lines may be referenced by symbols (FIRST, LAST, etc.). Line number design is decimal, which allows insertions of up to 999 text lines between 2 existing lines. Full screen management permits fast review and modification of entire screens for 3270-compatible terminals.

Data Set and Catalog Management. Text may be saved permanently on disk as sequential data sets or partition data set (PDS) members. Input and output records are variable length up to 133 characters. The data sets may be saved, retrieved, cataloged, recataloged, uncataloged, or scratched. PDS libraries may be compressed automatically to save space and access time. Directories and VTOCs may be examined for private as well as public libraries. Library access is security protected on several levels, and data set integrity is maintained during updates.

Remote Job Processing. Job streams may be prepared and submitted from a terminal. The user may inquire about the status of any job in the system using INTERACT commands. HASP/JES commands are supported for print, route, purge, hold, release, cancel, and alter functions. The output of any job on the system spool files may be retrieved at the terminal; no special JCL is required when submitting the jobs. Output may be retrieved as many times as desired. Output is available until the user instructs INTERACT either to purge it or to print it at a central or a remote location. When job output is retrieved at a terminal, it is numbered automatically, and all of the text editing commands may be used to examine the output. Access to jobs, like access to data sets, is also security protected.

Word Processing. Text may be formatted easily by the document processing feature of INTERACT. Facilities include automatic alignment and justification, centering, indenting, page composition and numbering, underlining, and headings and footings. INTERACT will even compose a table of contents and automatically generate the proper page references if requested. Special characters (such as superscripts and subscripts) may be included in offline listings. Form letters with variable text insertion are also handled. Formatting and reformatting of text is performed easily any number of times. Powerful list options allow the user to control spacing and skipping and to print multiple copies of documents with or without line numbers.

The Execute File Facility and the INTERACT Preprocessor. Using the execute facility, a sequence of INTERACT commands that perform a specific function may be executed by a single command. In addition, a subset of INTERACT commands coupled with the execute facility and the INTERACT preprocessor form a powerful programming language. Features provided include: variables, expression computation, assignment statements, data attributes, branching statements, and predefined functions for both numeric and character string processing. Data that may be manipulated include INTERACT commands, JCL, source code, input data, or any combination thereof. Any of these statements may contain variable symbols to signal points of parameter substitution. The INTERACT preprocessor will enter values in place of these points of substitution during execution. The values may be set by the user, or received as input from the terminal, and comments may be displayed at the terminal.

Cost Effectiveness

INTERACT has a total system approach that is simply unmatched. It offers an efficient means of dealing with text entry, text editing, and remote job processing. Significant operating efficiency and economy is provided in a number of ways:

- **Broad Application Range.** INTERACT provides computer services to many different types of users when they need it. Application programmers can submit jobs and retrieve output online at remote terminals without the delays of batch processing. System programmers can perform operating system activities, display operating system information, and issue privileged HASP/JES commands from a remote terminal. Typists have a full range of text editing commands for their use in creating and modifying text documents, which helps eliminate keyboard errors and increase revision speed. Users may have documents printed at a typewriter terminal or on a high-speed printer whenever copies are needed.

- **Improved Turnaround.** INTERACT's remote processing ability eliminates many of the problems associated with batch runs. No more waiting on printer backlogs — job output can be retrieved at a terminal to determine whether the job has run successfully. For the programmer, this means less time spent waiting for resources, more time spent doing productive work.

- **Increased Productivity.** INTERACT provides a time-saving approach to text creation and manipulation, whether it be for programming activities or document processing. Users can make extensive changes quickly, using facilities that permit global modifications throughout an entire file or several files. Tedious and time-consuming editing can be performed quickly and effortlessly, thus freeing users for more important tasks. INTERACT helps relieve some of the load on computer operators. There are fewer card decks to read and punch and fewer printouts to be sorted and delivered when INTERACT's remote processing facilities are used.

- **System Efficiency.** INTERACT approximates block locations when searching files and reads and writes only what is necessary instead of processing the entire file. This provides significant access time reductions. INTERACT also uses an internal blank suppression technique to remove extra blank characters when storing data sets on disk, thus cutting down on disk space requirements. The blanks are not lost — INTERACT remembers where to put them when the files are retrieved.

- **Minimum Operating Considerations.** INTERACT is available over a full range of operating systems: OS/MFT and MVT, VS1, and VS2 (SVS/MVS). Upgrading to another operating system or machine requires no database conversion or re-entry. INTERACT does not disrupt current system configurations or operating schedules. It may be run alongside batch processing and other online systems. INTERACT supports standard terminals: local and remote 3270s, 2741s, TTYs, or equivalent types.

- **Comprehensive Accounting Facilities.** Accurate accounting information is automatically generated for the SMF data set and for INTERACT-generated reports. An installation may choose to use standard SMF accounting programs or write its own. Accounting information is gathered for disk, page, card, and line counts; CPU time; elapsed time; run times; etc.

INTERACT can enhance the total productivity of an installation while providing substantial savings, including reduced paper and card costs and fewer new equipment requirements.

Text Entry and Editing

The user signs on the INTERACT system and is assigned a work space (file) for holding text to be created or manipulated. An active file can also be used to hold input and output when running jobs remotely. All of the text editing commands will work on an active file. Using line numbers assigned by the system, the user may issue any of the following commands to manipulate text:

COLLECT	Save and number text in an active file
LIST POINT	Display all or part of an active file online or print it offline
PUNCH	Punch all or part of an active file to cards
CHANGE EDIT MODIFY	Insert, replace, or delete single characters or character strings in specific columns, in single lines or line ranges, or in an entire file
INSERT	Insert lines of text between existing lines
REPLACE	Replace existing lines of text with new text lines
DELETE	Erase entire lines of text
COPY MOVE	Copy lines from within an active file or from an external file to a location within an active file
COMPARE	Compare the text in two sets of consecutive lines, existing in one of more active files, line by line
NUMBER	Renumber all or part of an active file
CLEAR	Erase an active or execute file, the tab settings, or the data set name fields
LOGON LOGOFF	Establish or terminate terminal sessions on INTERACT

SPECIAL EDITING FEATURES

Full Screen Management. This feature allows a user working at a 3270 terminal to enter an entire screen of data into an active file with a single depression of the enter key. The ability to manipulate full screens of data at once makes text editing faster and greatly reduces I/O processing.

Multiple Active Files. This permits more than one file to be edited simultaneously. This is useful in those situations where the user wants to make changes in one file while looking at another file.

Data Set and Catalog Management

When the task of entering text into an active file has been completed, a user may save a copy of the text on disk as an external file. Once saved, it may be brought back into an active file any number of times for examination or change. The commands used to create, access, and manage data sets as external files are:

SAVE	Copy all or part of an active file onto external file space
USE COPY	Read an external file or any online sequential disk data set or PDS member into an active file
SCRATCH	Scratch a specified external file
CATLG RECATLG UNCATLG	Catalog, recatalog, or uncatalog data sets on the system catalog
CONDENSE	Submit a job that will compress a specified PDS
GIVE	Allow one group member to give INTERACT disk space to another group member
PROTECT	Associates a password with a data set so that anyone using the data set must specify the password in order to access it. Once created, passwords may be deleted or changed.

Remote Job Processing

A user at a remote terminal may run any job that can be processed in the conventional batch processing mode. A user may also check the status of any job in the central computer's job stream. The following set of commands are used to interface with the operating system in performing these functions:

RUN	Submit all or part of an active file into the job input stream of the central computer
LOCATE	Inquire about the status of any job in the system
FETCH	Retrieve job output at a terminal
HOLD RELEASE	Move jobs from a wait queue to a hold queue or vice versa
PRINT	Release output from the hold queue for printing or punching
CANCEL PURGE	Cancel executing jobs or suppress output of jobs in an input or output queue
ROUTE	Change the device to which the job's output will be sent
TO	Send a message to another user or to the central computer operator
MAIL	Send all or part of an active file to another user or to the system manager

The Execute Facility

INTERACT has an Execute Command Language which consists of all the standard INTERACT commands plus a set of logic and arithmetic statements. This language is extremely powerful, yet easy to learn, and is available to all users of the system.

Execute programs may be prepared by the user to perform a specific function and can be executed from any INTERACT terminal. Any of the INTERACT commands may be included in an execute program. These programs may be saved and retrieved or executed by a single command.

Execute routines are often written to handle frequently used functions. For instance, an execute routine can be used to build JCL in an active file with one simple command. One might be created to search, retrieve, and compile information from personnel files, internal telephone directories, customer lists and the like.

Several INTERACT commands are designed specifically for the execute facility. These commands provide a high-level programming language when used with the INTERACT preprocessor. With this facility, a user can build an execute program that will receive input from the terminal by prompting the user for variables, verify the input, control the sequence and content of its statements using decision-making capabilities, and display messages to the terminal during processing.

Execute routines can also be written to perform operating system functions for personnel who do not have prior knowledge of programming. An execute program can perform a task in minutes that would normally require hours or days. For performing many routine functions quickly and easily, the execute facility is unmatched for convenience.

EXECUTE COMMANDS

READ	Obtain input from an active file or directly from the user
IF	Evaluate expressions, and provide conditional execution ability
EXECUTE	Load and execute a file, or cause unconditional branching within an execute program
COMMENT	Display messages to the terminal
RESTORE	Move the current execute file to an active file
SAVE EXEC CLEAR EXEC LIST EXEC PUNCH EXEC POINT EXEC MAIL EXEC COPY EXEC RUN EXEC	Save, erase, display online or offline, punch, submit to another user or to the central computer, or copy to an active file all or part of the user's execute file

SET OR SHOW:

RETURN	Provide "branch and link" return ability
VALUE	Set or show values of variables
ESCAPE SKIP RESCAN	Determine where points of substitution should be made or bypassed, or reset the maximum number of times that the preprocessor will scan the text.
EXEC	Allow messages to the terminal to be shortened or eliminated

Word Processing

The INTERACT word processing feature has the ability to assist the user in the preparation of high quality letters, documents, or other forms of text. With the INTERACT word processing feature, the user will not only save time but the quality of all documents produced will be superior to other forms of preparation.

Commands exist to center text and to format text with even left or right margins. Document processing commands perform automatic table of contents preparation, page composition and numbering, and handling of special characters such as superscripts and subscripts.

By using a combination of INTERACT commands, the user can process high volume applications such as letter writing or label generation with a very small amount of manual effort. It is possible that a single word processing application could justify the cost of INTERACT. A partial list of the word and document processing commands is:

ALIGN JUSTIFY	Reformat lines to contain no more than or exactly the number of characters specified for the line length.
CENTER	Center lines of text within a line length.
PAGINATE	Create pages of text with a user specified number of lines per page.
TITLE	Provide a heading for text pages.
FOOTER	Provide a footing for text pages.
SECTION	Delineate a section title to be included in the table of contents and specify whether or not it is to be underlined.
INDENT	Specify the number of leading spaces to be added to each line and the format of paragraphs to be used.

Defining Session Parameters

INTERACT allows the user to set values for some of the command parameters and terminal settings. Once a command parameter is set, it may be omitted from any INTERACT command in which it appears. Terminal settings (tabs, line length, etc.) also may be set differently for different types of text entry. In addition, the user may ask INTERACT to show the current value of any of these predefined parameters. Values can be set or shown for the following parameters:

SET or SHOW BACK NOBACK	Specify whether backspaces will erase characters or will be retained in the text.
CURRENT	Set the current line number.
DELTA	Set the line number increment.
FASTLIST SLOWLIST	Set the speed of listing at a terminal by allowing or disallowing the use of tabs.
NAME PREFIX USER GROUP MEMBER LIBRARY	Set default values to be used in forming an INTERACT data set name.
KEYWORD	Set a user's keyword for security measures.
LENGTH TABS UPPER UPLOW	Set for a terminal: line lengths, tab positions, and uppercase or upper/lowercase character recognition.
TERSE VERBOSE	Determine if INTERACT will issue complete or abbreviated responses.
MODE	Set either automatic or nonautomatic retry for commands that fail.
NOTIME TIMEOUT	Determine if users who have been inactive for 20 minutes are to be signed off.

VOLUME	Set a default disk volume identification.
FUNCTION	Assign a function key to an INTERACT command.
FILE	Set the attribute of a file.

In addition to setting and showing predefined parameters, INTERACT allows the value of certain system parameters to be displayed, as follows:

SHOW CATLG DIRECTORY DSNAMES	List data set names or index pointers in the system catalog or a specified volume, or members in a PDS.
TIME DATE	Display the current time and session time, or time, date, and Julian date.
COLUMNS	List column numbers in a line.
FIRST LAST NEXT, etc.	Display the value of the specified symbolic line number.

COUNT
USER(S)
LINE(S)
	Display a variety of information regarding users and lines on the system.

FREE IDLE MINFREE	Give a count of free phone lines (i.e., INTERACT lines not in use) and list their INTERACT line numbers.
SPACE TRACKS	Display tracks available for all users or a single user on specified volumes.
JOBNO	Display the numeric portion of the job name assigned by INTERACT.
VOLUME(S)	List all disk packs accessible to INTERACT.
MESSAGE	Display the contents of the current INTERACT system message.
SIZE	Display line numbers of lines over, under, or equal to a specified number of characters in length.
PAGES	List the number of pages of text in an active file.

Implementing INTERACT

Installation. INTERACT is easy to install — a typical installation takes less than 60 minutes. No special system generation is required — there are NO operating system modifications. INTERACT is available under various license arrangements designed to fit each installation's financial requirements, and is fully maintained and continually enhanced. INTERACT is currently in use at installations throughout the United States, under a variety of operating systems.

Training. Users can learn how to use INTERACT in a short period of time. Manuals covering basic operations, advanced techniques, and the execute facility are supplied with the system. Documentation is most effective when used with a terminal so that the user can gain hands-on experience. This approach allows users to become acquainted with INTERACT at a time that is most convenient to them, and when access to a terminal is available.

Documentation. Customers receive complete and updated documentation for all the features of the INTERACT system.

Software/Hardware Requirements. INTERACT will run as a problem program on any of the following operating systems:

OS/MFT HASP
OS/MVT HASP
VS1 JES1
VS2 (SVS) HASP
VS2 (MVS) JES2

INTERACT is written in IBM Assembler Language and requires one region/partition for operation. INTERACT has its own input/output routines and uses EXCP-level programming.

The minimum system, model, and storage size required by INTERACT is an IBM System/360 Model 50 with 512K of real storage, or an IBM System/370 Model 135 with 512K of real storage. A typical system supporting 16 users requires approximately 256K of virtual storage.

Generally, all of the equipment used by INTERACT is already required by the operating system. Specific requirements are:

- **Direct access storage.** Disk space is required for storing the INTERACT program and for text stored by the user. Devices used by INTERACT can be IBM 2314, 3330, 3330-11, 3340, 3350, 2305-1, 2305-2, or equivalent devices. Approximately 10 megabytes of storage are required for the INTERACT system.

- **Tape Drive.** A 9-track tape drive is required for loading the installation tape.

- **Terminals.** Communications may be supported via local 3270 display terminals, 2741s or TTY-compatible terminals through an asynchronous port of a 270X or 370X control unit; remote 3270 display terminals attached through a bisynchronous port of a 270X or 370X control unit; 3767 terminals in TTY or 2741 mode; or equivalent devices. The maximum number of terminals that can be attached is theoretically 32,000; however, the amount of main storage available is also a determining factor. A major feature of INTERACT is its ability to interface to terminals through a variety of TP monitors and access methods such as SHADOW, INTERCOMM, and VTAM.

- **High Speed Printer Options.** If uppercase and lowercase printing is desired from the computer's line printer, an IBM 1403 or 3211 printer equipped with the universal character set feature and the TN printing arrangement, or a 3800 Printing System (or an equivalent device) must be used.

In Conclusion...

The INTERACT system optimizes computer resources. The combination of features in INTERACT allows text entry and manipulation to be made with exceptional speed and ease. INTERACT's remote job processing ability relieves many of the problems associated with batch runs — less time is spent getting information into and out of the computer.

INTERACT is compatible with existing software systems, without operating system modifications, and can make a computer center vastly more productive. INTERACT not only provides increased productivity, but also provides greater job satisfaction.

DOCUMENTATION COMMENT FORM

Cullinane Corporation

INTERACT Summary Description
March 1979

Use this form to express your views concerning this publication. All comments and suggestions are both welcome and appreciated.

Please do not use this form to report system problems or to request additional publications, new release materials, etc. System problems should be reported on Cullinane's standard Exception Report form, or in a separate letter addressed to the attention of the appropriate support group. Requests for published materials should be addressed to the attention of Distribution.

SENDER (fill in only if you wish):

_______________________________ _______________________________
(name) *(position)*

____
(company)

____
(address)

____
(city, state, zip)

COMMENTS:

Areas of comment are general presentation, format, organization, completeness, clarity, accuracy, etc. If a comment applies to a specific page or pages, please cite the page number (s).

____

____

____

____

____

____

____

____

____

Thank you for your response.

Order Number: TDIN-110-60

cut or fold along this line

fold

Cullinane Corporation

Specialists in Data Management Software

First Class
Permit No. 41947
Wellesley, Mass.

Business reply mail. No postage necessary if mailed in the United States.

Postage will be paid by:

Cullinane Corporation
20 William Street
Wellesley, Massachusetts 02181

Attn: Documentation

fold

Cullinane Corporation

DOCUMENTATION COMMENT FORM

INTERACT Summary Description
March 1979

Use this form to express your views concerning this publication. All comments and suggestions are both welcome and appreciated.

Please do not use this form to report system problems or to request additional publications, new release materials, etc. System problems should be reported on Cullinane's standard Exception Report form, or in a separate letter addressed to the attention of the appropriate support group. Requests for published materials should be addressed to the attention of Distribution.

SENDER (fill in only if you wish):

______________________	______________________
(name)	*(position)*
______________________	
(company)	
______________________	
(address)	
______________________	
(city, state, zip)	

COMMENTS:

Areas of comment are general presentation, format, organization, completeness, clarity, accuracy, etc. If a comment applies to a specific page or pages, please cite the page number (s).

____

____

____

____

____

____

____

____

____

Thank you for your response.

Order Number: TDIN-110-60

cut or fold along this line

fold

Specialists in Data Management Software

First Class
Permit No. 41947
Wellesley, Mass.

Business reply mail. No postage necessary if mailed in the United States.

Postage will be paid by:

Cullinane Corporation
20 William Street
Wellesley, Massachusetts 02181

Attn: Documentation

fold

IBM SPF

VM/CMS - 3270 DISPLAY SUPPORT AND STRUCTURED PROGRAMMING FACILITY (SPF/CMS) 5748 - XT3

VM/CMS - 3270 Display Support and Structured Programming Facility (SPF/CMS). Program Number (5748-XT3) is functionally equivalent to Version 2.2 of the SPF/TSO Program Product.

The two Program Products are fully compatible in display formats and operation, except for those features which are explicitly oriented to the VS2/TSO or VM/CMS environment. SPF/CMS also includes a utility which permits transfer of SPF program libraries from VS2/TSO to VM/CMS.

HIGHLIGHTS

SPF/CMS is a program development tool designed to take advantage of the characteristics of IBM 3270 display terminals, and to increase programmer productivity in the VM/CMS environment. SPF supports both structured and conventional programming techniques. It can be used either by an individual programmer, or by many programmers working together on a project. SPF features which increase programmer productivity and simplify operation include:

- Display presentations and menus which prompt the user, reduce keystrokes, and minimize the opportunity for error.
- Support for multi-level programming libraries, automatic collection of library activity statistics, and printing of library contents.
- Full screen, context editing which allows additions and changes to multiple lines in a single interaction.
- Simple one-character edit commands for inserting, deleting, duplicating, or rearranging lines of source data.
- Forward, backward, and sideways scrolling of source data or listings, plus the ability to locate information by character string or line number.
- Split screen, allowing two SPF functions to be performed independently on the same display terminal.
- Use of program function keys for frequently performed SPF operations and commands.
- Menu-driven utilities for specification, maintenance and current status display of libraries and files.
- Menu interface to standard language processors (compilers and assembler) for execution in the foreground or CMS batch machine.
- Document preparation support, including text editing features and a menu interface to the SCRIPT/VS Document Composition Facility.
- Hardcopy log summarizing significant user actions during the session.
- Online tutorial for instruction and reference - especially valuable for the occasional or novice user of SPF.

SPF operates as a VM/CMS application and is invoked simply by entering the command "SPF". It operates on a 24-line, 32-line, or 43-line IBM 3270 display station, equipped with either 12 or 24 program function keys.

DISPLAY FORMAT

SPF uses four basic types of display presentations:

- **Option Selection Menus** - where the user selects from a list of options by typing a number and pressing the ENTER key.
- **Parameter Entry Menus** - where the user supplies parameters by filling in labeled fields. In many cases, SPF displays default values, based on what the user last entered. The user may overtype the displayed defaults.
- **Member Selection Lists** - for displaying a list of members in a MACLIB or SPF library. The user may select a member by entering a one-character code in front of the appropriate member name.
- **Data Displays** - for displaying source code or listings for editing and browsing. The first two lines of the display include a title, message area, command input field, and scroll amount field. The remaining lines contain the data.

PROGRAM ACCESS AND FUNCTION KEYS

The program access (PA2) and the 12 or 24 program function (PF) keys are used to request commonly used or special SPF operations, including:

- RESHOW - to redisplay the contents of the screen.
- SCROLL - to move the screen window up, down, left, or right by the amount shown in the scroll amount field
- SPLIT SCREEN - to enter split-screen mode or to adjust the locations of the split based on the current cursor position.
- PRINT - to obtain a hardcopy snapshot of the current screen image.
- HELP - to obtain additional information about an error message or general information about SPF commands and options.
- END - to terminate the current operation and return to the previous display or menu.

Users may rearrange the assignment of SPF-defined PF key functions, and may equate additional PF keys to edit and browse commands.

MAJOR FUNCTIONS

The major functions provided by SPF are:

- **SPF PARMS** - to specify terminal type and user options.
- **BROWSE** - to display source data and output listings.
- **EDIT** - to create or change source data.
- **UTILITIES** - to perform SPF utility functions.
- **FOREGROUND** - to compile, assemble, or debug.
- **BATCH PROCESSING** - to compile or assemble via a CMS batch machine.
- **CP/CMS** - to enter a CP or CMS command, or EXEC.
- **TUTORIAL** - to display information about SPF.
- **EXIT** - to terminate SPF.

SPF PARMS

The SPF parms function allows the user to specify terminal type, mono/dual case, number of program function (PF) keys, default options for list and log files, and PF key definitions.

BROWSE

The browse option allows the user to display source programs, output listings, test data, etc., stored in SPF libraries, MACLIBs or DASD resident sequential CMS files with the following characteristics:

- Fixed or variable record formats.
- Records with or without printer control characters.
- Records with logical record length up to and including 32,767 bytes.

The screen window for browsing is 22, 30 or 41 lines deep depending upon terminal type. (Two lines are reserved at the top of the screen for title information, messages, command entry, and display of the scroll amount.) Four-way scrolling is available via the scroll PF keys. In addition, FIND and LOCATE commands may be used to scroll to a designated character string, line number, or symbolic label.

EDIT

The edit option allows the user to create, display, and modify source data (program code, test data, documentation, etc.) stored in SPF libraries or DASD-resident sequential CMS files with the following characteristics:

- Fixed or variable record formats.
- Records with or without printer control characters.
- Records with logical record lengths of up to and including 255 bytes but not less than 10 bytes.

The edit display is similar to browse except that each line consists of a six-column sequence number field followed by a 72-column data field. Four-way scrolling is available via the scroll PF keys.

To modify information on the screen, the user simply moves the cursor to the desired location and enters the new information. Several lines may be modified before pressing the ENTER key. Lines may be deleted, inserted, shifted left or right (for indentation changes), duplicated, or rearranged by overtyping the sequence number fields with line commands consisting of one or two characters (e.g., D for delete, I for insert, M for move). In general, several line commands as well as data modifications may be typed before pressing the ENTER key.

In addition, primary commands may be entered at the top of the screen to find and change designated character strings, to control sequence numbering and character translation, to submit the data to the job stream, to save the edited data, or to cancel without saving.

UTILITIES

The SPF utility function allows the user to select one of the following options:

- **Library Utilities** - to compress a MACLIB, to print a file, or to delete, rename, or print selected members of a MACLIB or SPF library
- **File Utility** - to specify a new SPF library, to rename or delete a file or SPF library or to display file or library information.
- **MOVE/COPY** - to move or copy an entire file or selected members of a MACLIB or SPF library
- **Project Utility** - to print or display information about SPF libraries.
- **Reset SPF Statistics** - to create, update or delete statistics for an SPF library, and optionally reset sequence numbers.

SPF/CMS (cont'd)

- **Hardcopy Utility** - to initiate printing or punching of a sequential CMS file or member of a MACLIB or SPF library.
- **Retrieve Utility** - to transfer SPF libraries which have been dumped to tape (via IEHMOVE) from SPF/TSO to SPF/CMS.
- **SCRIPT/VS Utility** - to allow formatting, displaying and printing of documentation maintained in SPF libraries or sequential CMS files. Use of this feature requires that the SCRIPT/VS Document Composition Facility be installed.

FOREGROUND PROCESSING

The foreground option provides an interface with standard language processors for foreground compilation, assembly, CMS loading, or debugging of programs stored in SPF libraries.

BATCH PROCESSING

The batch processing option provides an interface with standard language processors for CMS batch machine compilation or assembly of programs stored in SPF libraries. For other batch machine jobs, the CP SPOOL and PUNCH commands or some EXEC may be entered via the CMS command option (primary option 6).

CP/CMS COMMAND

The CP/CMS command option displays a menu on which any CP or CMS command or EXEC may be entered without leaving SPF.

TUTORIAL

The SPF tutorial provides immediate online reference and instruction on how to use SPF. It may be viewed sequentially from beginning to end, or randomly by selecting topics from a table of contents or index. It may be entered from the primary option menu, or from another SPF option by pressing the Help PF key.

SPECIFIED OPERATING ENVIRONMENT

Machine Requirements

The computer system requirements are the same as needed for VM/370; that is an IBM System 370 model 135 and up with at least 384K real storage, or a 4331 or 4341.

The storage requirements for the user's VM machine will vary depending upon the size of the files being edited and the extent that "split-screen" will be used. The SPF programs are reenterable and should be placed in a discontiguous shared segment area. This will reduce the storage requirements for user machines and should also improve performance when there are multiple SPF users online.

The minimum virtual machine size is 512K if SPF resides in the discontiguous shared area. If SPF does not reside in the discontiguous shared area, a virtual machine size of 768K is recommended.

Terminals

SPF supports the following IBM 3270 Display Stations:

- 3275 Model 2;
- 3276 Models 2, 3, and 4;
- 3277 Model 2 (local or remote attachment);
- 3278 Models 2, 3, and 4 (local or remote attachment).

The following keyboards are supported:

For 3275 or 3277 display stations:

- 78 Key EBCDIC Operator Console (feature #4632);
- 78 Key EBCDIC Typewriter (feature #4633);
- 78 Key ASCII Typewriter (feature #4635);
- 78 Key EBCDIC Typewriter/APL (feature #4638 for 3277 only), when operated with APL switch OFF;

For 3276 or 3278 display stations:

- 75 Key EBCDIC Typewriter (feature #4621);
- 75 Key ASCII Typewriter (feature #4624);
- 87 Key EBCDIC Typewriter (feature #4627);
- 87 Key ASCII Typewriter (feature #4628).

The standard character set (94 graphics plus blank and null) is supported on 3276 and 3278 display stations.

The audible alarm (Feature #1090) is supported, but not required. Installation of the audible alarm feature is strongly recommended to enhance usability. The alarm is sounded by SPF whenever a warning or error message is displayed.

Programming Requirements

This program product is released to work with VM/370 Release 5 and subsequent releases and modification levels unless otherwise stated. Prerequisite is either the VM/ System Extension program product, (5748-XE1) or the VM Basic System Extensions program product, (5748-XX8).

SPF/CMS is written in PL/S* and translated into Assembler Language

*Programming Language/System (PL/S) is an IBM proprietary language.

CUSTOMER RESPONSIBILITIES

The customer must have installed the prerequisite programs and the appropriate processing programs described under compatibility for use of the SPF foreground and batch processing functions.

COMPATIBILITY

SPF operates as a CMS command. SPF provides menu interfaces with the following IBM processing programs for foreground and batch execution:

VM/370 Assembler	
COBOL-OS/VS Compiler and Library	5740-CB1
COBOL Interactive Debug (foreground only)	5734-CB4
FORTRAN IV G1 Compiler	5734-FO2
FORTRAN Interactive Debug (foreground only)	5734-FO5
PL/I OS Checkout Compiler	5734-PL2
PL/I OS Optimizing Compiler	5734-PL1

All the program-numbered items above can be ordered separately under IBM program product licensing agreements. The appropriate processing programs must be installed to use the SPF foreground and batch processing functions.

SPF also provides a menu interface to SCRIPT/VS to allow formatting, display, and printing of text maintained in SPF libraries or CMS sequential files. Use of this feature requires installation of the following IBM program product:

Document Composition Facility (SCRIPT/VS) (with the Foreground Environment Feature)	5748-XX9

General Documentation: Available from Mechanicsburg.

General Information Manual
Licensed Program Design Objectives

Program Service: See PP Index.